Privacy-Preserving HIE by Multi-Party Deterministic Noise Generation

Xi Liu Yuzhe Tang Katchaguy Areekijseree
Department of EECS, Syracuse University, NY, USA,
{xliu314, ytang100, kareekij}@syr.edu

I. Introduction

In the big-data age, as human activities are increasingly captured in a digital form, sharing the digital personal data while facilitating information accessibility for new applications, poses a significant threat to the personal privacy. In the domain of the health-care, a federated database infrastructure called Health Information Exchange or HIE recently emerges [8], [11], [2], [3], with the goal of facilitating the exchange of electronic medical records or EMRs[1] among mutually untrusted hospitals and under the governance of HiPAA alike data protection laws.

The core component of an HIE is a locator service that bridges the producer and consumer hospitals [1], [8], [7], [10]: Instead of asking the patient to carry her own case history, an unreliable approach used in the paper-based healthcare,[2] the HIE locator stores and serves the EMR location meta-data on a public cloud for reliable producer discovery with 24/7 availability.

The fact that the EMR meta-data is stored in the public cloud raises privacy concerns. On the one hand, the third-party cloud operated by profit-driven companies is untrustworthy and potentially malicious. On the other hand, the EMR location meta-data (i.e., "given a patient, which hospitals store her EMRs") is privacy-sensitive and could reveal very personal matters [51]; e.g. the fact that "Robin Williams" visited a rehabilitation center could reveal his drug history, and Mr. Tiger Woods being hospitalized revealed his car wreck and affair in 2009. From a legal perspective, hospitals sharing raw EMR meta-data with the public cloud violates the data-protection laws, HiPAA [4], thus calling for privacy-preserving data sharing.

Towards the goal, we model the construction of an HIE locator as a multi-party noising problem [26], [28], [45] where multiple parties jointly publish results with noises built-in, while preserving individual privacy. Two unique challenges come about when applying existing techniques to the privacy-preserving HIE:

- Serving-time privacy assurance: Privacy needs to be preserved at the locator serving time when the target EMRs are located prior to sharing. Existing randomized noise generation [27], notable for differential privacy in statistics aggregation, is not well-suited for serving-time privacy-preservation – The random noises introduce uncertainty for privacy assurance. Injecting noises in a non-randomized (or deterministic) fashion is essential to the privacy assurance in the noise quality (indistinguishability) and quantity. To ensure the noise indistinguishability, it entails modeling the attack's background knowledge.
- Construction-time privacy and efficiency: The deterministic noising entails complex computation and thus general-purpose MPC [62], [42]; the special-purpose MPC alternatives can neither provide assured privacy preservation [59], [35] nor support generic computation [46], [22]. The key challenge for putting the existing general-purpose MPC techniques in practice is their high performance overhead [44], which could be prohibitive when applied to the HIE big-data scenario, where there are typically a large volume of input data (e.g., the big medical data), a large number of parties (e.g., hundreds of hospitals in a state-wide HIE and thousands in a nation-wide HIE) and complex computation logic (e.g., required by the deterministic noising and background knowledge modeling).

To tackle the two challenges, we propose $\mathsf{P}^3\mathsf{I}$ (privacy-preserving patient index) to serve HIE locator requests and to allow configuration knobs by expected noise level. For $\mathsf{P}^3\mathsf{I}$ construction, we propose MPDN, a multi-party deterministic noising protocol that preserves the construction-time privacy under the configuration. Our approaches are:

- For assured serving-time privacy, MPDN models the background knowledge and makes selected noises indistinguishable by a top-k similarity computation. The top-k computation strikes a balance between the practical computation complexity and the needs to counter the background knowledge attacks of exponentially growing possibilities. Although our background-knowledge modeling is not exhaustive, we claim our approach is realistic and practical by our evaluation on real-world datasets.
- MPDN optimizes the performance of applying MPC

[1]EMR in the real world has a standard format, continuity care documents,
http://www.hl7.org/implement/standards/product_brief.cfm?product_id=6.

[2]Consider the patient is forgetting or, in emergencies, is sent to the hospital unconsciously.

to the P^{3}I construction, through pre-computation on background knowledge. Our insight is that the background knowledge is public information, and the related computation can be made without MPC. To construct P^{3}I, MPDN separates the public background-knowledge computation from the private computation on sensitive EMR meta-data. It then applies expensive MPCs only to the latter. In addition, we perform system-level optimization by exploiting the innate data-level parallelism and using the GPGPU (general-purpose GPU) to speed up the pre-computation.

The contributions of this work are listed as following:

- We model the problem of serving HIE locator in a public domain for privacy-sensitive EMR exchange. We formulate the privacy definition and background knowledge attacks, both tailored to the setting.
- We present a holistic solution to tackle the privacy-preservation. We propose deterministic noising with assurance in privacy and resilience to background-knowledge attacks.
- We optimize the MPC performance for locator construction in large-scale HIE networks. The proposed optimization works by pre-computation on public background knowledge and by leveraging GPGPU to exploit the data-level parallelism. Through implementation using real-world software and evaluation in a geo-distributed setting, we demonstrate the performance optimization by more than an order of magnitude.

II. Problem Formulation

This section formulates the privacy-preservation problem of the HIE locator. We formally describe the general system model for an HIE locator, and analyzing the security of existing privacy-preserving locator solutions, in a way to formulate our background-knowledge based attack model and the attack-mitigation goals.

A. System Model

We present the system model through describing the life cycle of sharing an EMR from its producer hospital to the consumer hospital. This process involves four principles: hospitals participating in an HIE, a patient, physicians, and the HIE locator. In particular, the HIE locator bridges an EMR consumer (e.g. a physician) with an EMR producer (e.g. a hospital); In the real world, it can be set up by running OpenEMPI software [10] on an Amazon AWS alike public cloud. The process of sharing an EMR is divided into three stages:

A. EMR generation, when the EMR is produced as a result of the patient's treatment or diagnosis; the EMR's storage is delegated to the producer hospital where the treatment occurs. The interface is ①. H_i.DELEGATEEMR(p): patient p delegates her personal EMR to the visited hospital, H_i. There is a natural mutual trust between the patient and the hospital.

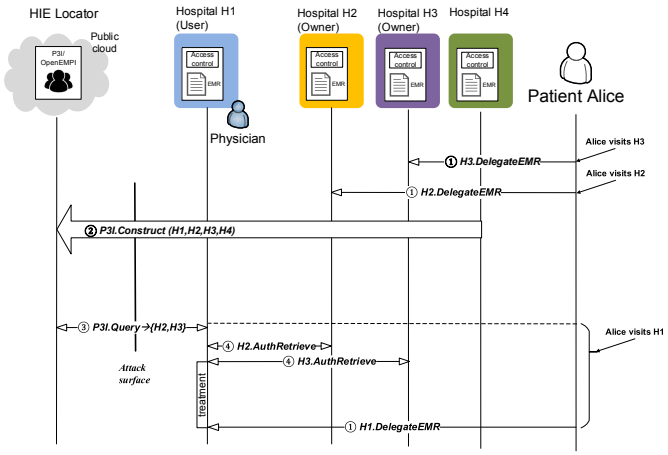


Fig. 1: Life cycle of sharing an EMR in the HIE: The figure illustrates the scenario of patient Alice's visit to hospital H_3 when her physician needs to retrieve her past EMR at hospitals H_2 and H_3. Different background colors mean different trust relationships, e.g., the physician in blue trusts hospital H_1 in blue, and hospital H_2 in yellow does not trust hospital H_1 in blue and HIE locator in gray.

B. Locator construction, when all the hospitals disseminate the EMR meta-data (i.e. "which hospital stores the EMR") to the public cloud to construct the locator. The formal interface is: ②. P^{3}I.CONSTRUCT($\{\forall H_i\}$). In here, hospitals do not mutually trust each other or the public cloud, as regulated by data-protection laws, HiPAA [4].

C. Locator serving, when the physician retrieves the past EMRs of a patient for better diagnosis. The physician is involved in a process of two interactions, the first one with the locator (③) and the second with the producer hospitals (④):

③. P^{3}I.QUERY(p) $\rightarrow$ $\{H_i\}$: A physician queries the P^{3}I[3] to locate the producer hospitals of patient p's EMRs, $\{H_i\}$.

④. H_i.AUTHRETRIEVE(p): Given the list of producer hospitals $\{H_i\}$ from P^{3}I, the physician contacts individual hospitals in $\{H_i\}$. Each hospital H_i will then authenticate the identity of physician and enforce access control before allowing her to retrieve the EMR.

In Figure 1, we use an example (i.e. patient Alice visits hospitals H_1, H_2, H_3) to illustrate the entire sharing process.

[3]We use P^{3}I here for illustration purpose; P^{3}I can be replaced by any alternative locator solutions.

Trust Model: In this work, we consider only patient privacy and data confidentiality, and assume the data/computation integrity has been taken care of. In our universe, the public cloud is not trusted by hospitals, patients or physicians. Among hospitals, there is no mutual trust unless the producer-consumer relationship is established during interaction ④. In interaction ①, patient trusts the hospitals she visits. There is also a mutual trust between a physician and her employer hospital.

1) Attack Model:

Attack surfaces: An attack surface refers to the interactions that cross the boundary between mutually untrusted parties. In our universe of the four interactions, the attack surface includes ②, ③, and ④ whereas interaction ① occurs between two trusting principles (the patient and the hospital being visited). An attack against interaction ④ can be mitigated through the standard cryptographic secure channel (i.e., set up by PKI and key exchange protocols [29]), and is out of our interest. We focus on the most challenging part, that is, formulating and mitigating the attacks on ② and ③.

The attack to ②, aiming at breaking "construction-time privacy", is to disclose the sensitive EMR location meta-data exchanged among hospitals during the construction. We assume the standard semi-honest model used in MPC [17], where the attacker plays a role of a participant hospital, honestly following the construction protocol yet being curious about the sensitive EMR meta-data received from remote hospitals. The construction-time privacy can be formally preserved with assurance by the general-purpose MPC.

The attack to ③, aiming at breaking "serving-time privacy", is to obtain the definitive EMR location information during the locator serving time. The attack can be modeled by a one-round game between the adversary and $\mathsf{P}^3\mathsf{I}$ as a challenger. Given a patient and hospital, the challenger presents the noised location information (elaborated in § III) to the adversary who then adaptively makes a probabilistic claim on the disease the patient is likely to have by exploiting related background knowledge. The adversary winning the game is defined to be that she can make the claim with probability significantly larger than she can without knowing the noised location information. To be more specific, we consider a flow of attacking actions through which the adversary exploits her initial knowledge (e.g., the noised location information, I) to get to the privacy-disclosing fact (e.g., the disease a patient must have).

2) Applicability of other protection techniques: Because of our trust model, many existing data-protection techniques are inapplicable to securing the HIE locator. Specifically, the access control and user authentication [39], [50] require a trusted computing base which the untrusted locator server can not provide, and the encrypted cloud storage [47] requires mutual trusts (in the form of shared keys) between the consumer physician and producers. Without the blessing of trust to distinguish legitimate searchers from attackers, the locator service preserves privacy by introducing fuzziness or noises into the query result while deferring the de-noise to the hands of trusted producers (i.e., in ④).

B. Existing Locator Solutions under Attacks

Attacking OpenEMPI: OpenEMPI [10] assumes a trusted cloud in storing the exact EMR location information (i.e. without noises) and in enforcing the access control for protection. This is a trust assumption we deem unrealistic. In our threat model, the OpenEMPI protocol discloses the serving-time privacy with certainty.

Attacking ϵ-PPI: ϵ-PPI (for Privacy-Preserving Index) [56] is the state of the art privacy preserving solution for HIE locator. It works by injecting randomized noises into the locator; the noises are randomly selected, in a way that is agnostic to the background knowledge. In addition, the actual amount of noises reach an expected level in a statistical sense, e.g., with 50% chance, without assurance.

We compare existing possible locator solutions in Table I under four metrics, privacy and efficiency at both serving and construction times. The search efficiency refers to the total number of hospitals returned from the locator, e.g., the broadcasting baseline performs badly in search efficiency because it always requires the physician to "broadcast" the retrieval request (④) to the entire network of HIE.

TABLE I: Comparison of locator solutions: $+/-$ represents the feature being supported/not supported. We use "Privacy" to refers to the assurance of privacy protection.

	HIE serving time		HIE construction time	
	Privacy	Efficiency	Privacy	Efficiency
$\mathsf{P}^3\mathsf{I}$	$+$	$+$	$+$	$+$
OpenEMPI [10]	$-$	$+$	$-$	$+$
PPIs [56], [57], [15]	$-$	$+$	$+/-$	$+$
Pure MPC construction	$+$	$+$	$+$	$-$
Broadcasting	$+$	$-$	$+$	$+$

C. Background-Knowledge Attacks

Background knowledge: In this work, the background knowledge considered (B) includes patient demographic information (e.g., age, gender, home address) and hospital profiles (e.g., specialties and location). Specifically, we represent the profile of each hospital by two metrics, a specialty vector and its geographic location (e.g. longitude and longitude). The specialty vector is a vector of ranking scores of the hospital in all specialty categories. The background knowledge we consider in this work is realistic and can be obtained from public data sources; for instance, the hospital profile in terms of location and specialties is public information available on the USNEWS

website [5]. And patient demographic information is from various online census datasets [9].

Defense by noising: $\mathsf{P}^3\mathsf{I}.\mathsf{Query}(p)$ presents false positive hospitals, serving as noises,[4] to obscure the identities of true positive hospitals which are privacy-sensitive to patient p. The definition of true/false positive and negative hospitals are the following.

Definition: For patient p, a hospital that she visited is defined to be a *truly positive* hospital, denoted by TP. The set of all true positive hospitals is denoted by $I^0 = \{TP\}$.

A hospital that the patient has never visited before is defined to be a *negative* hospital, denoted by N.

Definition: In the $\mathsf{P}^3\mathsf{I}$, a noise hospital is a hospital which the patient did not visit but the $\mathsf{P}^3\mathsf{I}$ claims that the patient visited. A noise hospital is a false positive and denoted by FP.

A hospital that appears positive in the $\mathsf{P}^3\mathsf{I}$ can be a true positive or a false positive, and the set of positive hospitals is denoted by $I = \{P\} = \{TP\} \cup \{FP\}$.

TABLE II: Notations

P: positive hospital	FP: true-positive hospital
N: negative hospital	TP: false-positive hospital
p: patient	$I^0 = \{TP\}$: true-positive hospitals

ϵ-privacy goal: Given the EMR location of a patient to a list of positive hospitals, one type of information leakage that is inevitable to achieving 100% search recall is that the adversary knows "all the true-positive hospitals are in the $\mathsf{P}^3\mathsf{I}$.QUERY result." Beyond that, we assume there is no direct information leaked on a patient's visited hospitals. For instance, the adversary does not know the total number of hospitals visited by a patient. Our privacy-preservation goal is to achieve ϵ-privacy for all considered attacks:

Definition: Given an attack that makes a probabilistic claim, ϵ-privacy is defined to be that the success rate of the attack is statistically upper bounded by ϵ.

Background knowledge attacks: The information flow of an attack is that the adversary can use the publicly available $\mathsf{P}^3\mathsf{I}$ to "reversely" infer the true-positive EMR location I^0, and then from I^0 (or I in some cases) infer the sensitive disease information of the patient. The information flow is formulated by the following ([] means optional):

$$I[\overset{a}{\mapsto} I^0] \overset{b}{\mapsto} disease \tag{1}$$

The background knowledge can facilitate the attack and can be used in two places in the attack information flow: 1. **Inferring disease in step** b: Knowing hospital specialties can assist in step b to infer the patient disease. The information flow by this attack is $I/I^0, B \overset{b}{\mapsto} disease$, where B represents background knowledge. 2. **Identifying noises in step** a: The background knowledge can be used in step a to distinguish true positive hospitals from the noises. The information flow by this attack is $I, B \overset{a}{\mapsto} I^0(\overset{b}{\mapsto} disease)$.

[4]Noise and false positive are interchangeable in this paper.

D. Domain-Specific Attacks

Under the above attack model, we consider three specific attacks. They are classified by attack information flow and background knowledge (as in Table III). Attack I does not rely on any background knowledge and aims at recovering true positive hospitals in step a. Attack II exploits background knowledge on the hospital specialty and aims at inferring patient diseases in step b. Attack III exploits various background knowledge on hospital and patient profile and aims at recovering true positive hospitals in step a. For different attacks, we present different top-k policies and analyze how the same ϵ-privacy assurance is achieved by $\mathsf{P}^3\mathsf{I}$.

TABLE III: Attacks: The considered attacks are classified by the type of background knowledge used in the attack and the information flow through which the adversary gets to the privacy-disclosing fact on "the patient's disease."

Attacks	Background Knowledge	Target in info. flow
Attack I	$\varnothing$	Step a
Attack II	Specialty	Step b
Attack III	Hospital profile, patient demographic info	Step a

TABLE IV: Modeling $\mathsf{P}^3\mathsf{I}$ data and background-knowledge: This table describes a scenario that involves one patient and five hospitals that appear to be positive in $\mathsf{P}^3\mathsf{I}$, $h_1, \ldots, h_5$. We consider two cases: one that all five hospitals are true positives, and one that among the five, h_2 and h_3 are the true positives. Background knowledge about hospital specialty implying patient gender, and geographic distance between hospital and patient home is presented. We also show the non-matching scores (m_e and m_f) on different background knowledge.

Hospitals	h_1	h_2	h_3	h_4	h_5
Case 1	TP	TP	TP	TP	TP
Case 2	FP	TP	TP	FP	FP
Specialty	Cancer	Rehab	Cancer	Woman's center	Rehab
Gender	F/M	F/M	F/M	F	F/M
m_e	1	1	1	0	1
Geo-distance	0.4	0.5	1	4	5
m_f	2.5	2	1	.25	.2

1) Attack I: In Attack I, the adversary randomly picks one hospital from $\{h_1, \ldots, h_5\}$ without any external knowledge, and makes a claim that the patient visited the hospital. The claim, if it's true, leaks the sensitive information (knowing a patient visits a rehabilitation center discloses her drug addiction problem). As illustrated by case 1 in Table IV, when all five hospitals are true positive, the claim is always true and any type-I attack always

succeeds.

2) Attack II: With the background knowledge of hospital specialties, the adversary can infer the health condition of the patient. The attack follows the information flow: $I, B \overset{b}{\mapsto} disease$.

The attack is successful when all positive hospitals end up with few specialties. Consider the extreme case that all positive hospitals are of the same type, say "rehabilitation centers". Then, no matter which hospital is true positive, the adversary can be certain that the patient must have an addiction-related problem. Note that this is different from attack I where the hospital in the claim must be true positive to have the attack to succeed.

3) Attack III: In Attack III, the adversary takes on step a to distinguish noise and true-positive hospitals by exploiting the knowledge on patient and hospital profiles. The patient profile in consideration is her demographic information such as home address, gender, age groups, etc. And the hospital profile includes the hospital's specialties, location, etc.

The attack works by the common knowledge on linking patients and hospitals. For instance, given a male patient, the adversary can easily determine that a woman's health center showing up in a $\mathsf{P}^3\mathsf{I}$ search result must be a false positive. Likewise, to a teenage patient, a hospital specialized in geriatrics is unlikely to be true positive. A hospital in the New York State is probably a false positive to a patient living in the State of Georgia. In general, the "non-matching" relationship through background knowledge can assist to reveal the identity of a noise hospital, thus improving the attack success rate. We formulate the relationship by non-matching score m.

Definition: Given the background knowledge B on patient p and negative hospital N, the non-matching score $m_B(p, N)$ measures the unlikelihood that the patient has visited the hospital. The non-matching score can also be expressed between a true positive hospital P and a negative hospital N, as $m_B(P, N)$.

Depending on the application scenarios, the non-matching score can take various values.

- **Exact-match**: The non-matching score takes a binary value, indicating whether the hospital matches the patient. In the previous example involving patient gender and hospital specialty, the non-matching score is 1 (0) when a male (female) patient and a woman's health center are considered. The implication of the non-zero score is that the woman's health center should not be chosen as a noise for a male patient.
- **Fuzzy-match**: The non-matching score takes values that are continuous. In the previous example involving a New York hospital and a patient in Georgia, the non-matching score is measured by the geographic distance between the two. The intuition is that the more distant a hospital is to the patient's location, the less likely there is a match between the two. The implication is that a hospital too far away from a patient should not be chosen as a noise for the patient.

III. Centralized $\mathsf{P}^3\mathsf{I}$ Construction

We now describe $\mathsf{P}^3\mathsf{I}$.CONSTRUCT($\{\forall H\}$), that is, how $\mathsf{P}^3\mathsf{I}$ is constructed with the required level of noises. In this section, we consider the centralized $\mathsf{P}^3\mathsf{I}$ construction by assuming a hypothetical authority that exists and is trusted by all hospitals. We will remove this assumption and present the secure and distributed $\mathsf{P}^3\mathsf{I}$ construction in the next section.

Asymmetric Deterministic Response: In the $\mathsf{P}^3\mathsf{I}$ construction, we allow a negative hospital to be published as a false positive. On the other hand, we do not allow false negatives, that is, a true positive hospital will always be published as positive. This rule, illustrated in Formula 2, leads to 100% search recall[5] and prevents any legitimate hospital from escaping the search result.

We call this publication primitive by asymmetric deterministic response (or ADR), reminiscent of the classic randomized response [60]. Comparing the randomized response, our ADR is *asymmetric* in that it treats binary input (N or P) differently or "asymmetrically", and is *deterministic* in that it flips input 0 based on certain deterministic conditions (described by top-k policies in § III-A).

$$\begin{aligned} ADR(N) &\rightarrow \begin{cases} P, \text{ chosen as noise by Algo. 1} \\ N, \text{ otherwise} \end{cases} \\ ADR(P) &\rightarrow P \end{aligned}$$

Top-k Algorithm: Given a patient whose location information to hospitals is I^0 (i.e. the list of visited hospitals), the $\mathsf{P}^3\mathsf{I}$ construction problem boils down to *noise generation*, that is, properly choosing a certain number of false positive hospitals. To a specific patient, the selection favors negative hospitals that may or may not be similar to the set of true positive hospitals (as will be discussed in § III-A). We thus define a hospital-to-hospital-set distance, $D(N, I^0)$, which measures the dis-similarity between a negative hospital N and the set of all true positive hospitals of a patient, I^0. The selection stops when certain condition is met. The top-k algorithm is illustrated in Algorithm 1.

Listing 1: top$k(I^0)$
Sensitive input I^0: true positive hospitals visited by a patient
Non-sensitive output $I = \{FP\} \cup I^0$: all positive hospitals

[5]Search precision in $\mathsf{P}^3\mathsf{I}$ is sacrificed for better privacy preservation. The implication of low search precision is that there are extra hospitals the record-searcher needs to contact.

$\{FP\} \leftarrow NULL$
$\{P\} \leftarrow I^0$
WHILE(stop−condition($\{FP\}$))
  Find h_{im} s.t.
    $D(h_{im}, \{P\}) = \min_{\forall i \notin I^0 \cup \{FP\}} D(h_i, \{P\})$
  $\{FP\}$.add(h_{im})
  $\{P\}$.add(h_{im})
RETURN $\{FP\} \cup I^0$

A. Mitigation and Security Analysis

1) Attack I: Mitigation and ϵ-Privacy: Recall that the top$k$ function in Algorithm 1 exposes two methods to configure: stop condition and distance definition; we call those two top-k policy. To mitigate Attack I, we use a top-k policy as below. Here, we re-use the notation of FP to denote the number of false positive hospitals. The distance is simply set to constant 1 which gives negative hospitals equal chances to be chosen as noise.

Attack-I mitigation:
- **Top-k stop condition**:
$$FP \geq TP(\epsilon^{-1} - 1) \quad (2)$$
- **Distance definition**: $D_I(N, \{P\}) = 1$

Proposition: P^{3}I can mitigate Attack-I with assured ϵ-privacy.

Proof: From the top-k stop condition in Equation 2, we can have the following.

$$\begin{aligned} & FP \geq TP(\epsilon^{-1} - 1) \\ \Rightarrow \quad & \frac{TP}{FP + TP} \leq \epsilon \end{aligned} \quad (3)$$

Given Attack I follows the information flow $I \overset{a}{\mapsto} I^0$, the success rate is:

$Pr(I^0|I) = \frac{Pr(\{TP\})}{Pr(\{TP\} \cup \{FP\})} = \frac{TP}{TP+FP} \leq \epsilon$

ϵ-privacy thus holds. ■

2) ϵ-Privacy under Attack II:

Straw-man by l-Diversity: Attack II might be mitigated by l-diversity [41] which in the P^{3}I context works by making a patient's diseases "anonymous" among l alternative diseases. However, l-diversity does not automatically lead to ϵ-privacy: While the former has restricts the number of different specialties, the latter restricts the number of negative specialties.[6]

ϵ-Privacy Assurance: The intuition of the protection is to choose enough false positive hospitals such that the false positive specialties suffice to bound the rate that the adversary can successfully pick a true-positive specialty. Formally, the top-k policy that assures ϵ-privacy under Attack II is described below. Here, FP_s (TP_s) denotes the number of false (true) positive specialties. A false positive specialty is a disease that a patient does not have but there is at least one positive hospital that is specialized in.

Attack-II mitigation:
- **Top-k stop condition**:
$$FP_s \leq \frac{TP_s}{1 - \epsilon^{-1}} \quad (4)$$
- **Distance definition**:
$$\begin{aligned} D_{II}(N, \{P_i\}) &= \|S(N) \setminus \cup_i S(P_i)\| \\ &= \| \cap_i [S(N) \setminus S(P_i)]\| \end{aligned} \quad (5)$$

The distance D_{II} between a negative hospital N and a set of positive hospitals $\{P_i\}$ is defined in Equation 5. Here, $S(\cdot)$ denotes the vector of specialties of a hospital. We use hamming distance to capture the difference ($\setminus$) between two specialty vectors. The distance definition favors the noises with different specialties from the true positive specialties. Thus, the number of false positive hospitals needed can be minimal, resulting in better search precision and performance.

Proposition: P^{3}I can mitigate Attack-II with assured ϵ-privacy.

Proof:

From the top-k stop condition in Equation 4, we can have the following.

$$\begin{aligned} & FP_s \leq \frac{TP_s}{1 - \epsilon^{-1}} \\ \Rightarrow \quad & \frac{FP_s}{FP_s + TP_s} \leq \epsilon \end{aligned} \quad (6)$$

Attack II follows the information flow $I, B \overset{a}{\mapsto} disease$ and is about recovering true positive specialties TP_s from the false positive ones FP_s. Then, the success rate is:

$Pr(disease|I, B) = \frac{FP_s}{FP_s+TP_s} \leq \epsilon$

Thus, the success rate is bounded by ϵ, hence ϵ-privacy. ■

[6]To be more specific, consider a counterexample against adopting l-diversity in P^{3}I. In Table IV (case 1), 3-diversity, is already there without any noises. Because all true-positives have totally three specialties, that is, Cancer, Rehab, Woman's center. However, given no noises, the success rate of Attack II can be as high as 100%, leading to a situation that achieves 3-diversity yet no protection in the sense of ϵ-privacy.

3) ϵ-Privacy under Attack III (Exact-match): We use the following top-k policy to mitigate Attack III with exact-match semantics.

Attack III mitigation (exact-match):

- **Top-k stop condition**:

$$FP \quad \geq \quad TP(\epsilon^{-1} - 1) \tag{7}$$

- **Distance definition**:

$$D_{III}(N, \{P_i\}) = \sum_i m_B(N, P_i) == 0?0 : \infty \tag{8}$$

Proposition: P^{3}I can mitigate Attack-III with assured ϵ-privacy in the sense of exact-match semantic.

Proof: Given Attack III follows the information flow $I, B \overset{a}{\mapsto} I^0$, the success rate is:

$$Pr(I^0|I, B) = \frac{TP}{TP + FP_0} \tag{9}$$

Here, we consider two types of false positive hospitals, the matching ones with zero-valued score (FP_0) and non-matching ones with one-valued score (FP_1). FP_1 can be distinguished by the adversary with background knowledge and thus should be discarded in accounting the success rate.

Our distance in Equation 8 is defined in such a way that any non-matching hospitals would lead to a distance of an infinitely large value, and thus will not be chosen by Algorithm 1. In other words, there are no non-matching hospitals that can be chosen as noises, that is,

$$\begin{aligned} & FP_1 = 0 \\ \Rightarrow \quad & FP = FP_0 + FP_1 = FP_0 \end{aligned} \tag{10}$$

Combining Equation 7, 9, 10, we can arrive at ϵ-privacy:

$Pr(I^0|I, B) \leq \epsilon$ ■

4) ϵ-Privacy under Attack III (Fuzzy-match): We use the following top-k policy to mitigate Attack III with fuzzy-match semantics.

Attack III mitigation (fuzzy-match):

- **Top-k stop condition**:

$$\frac{\sum_{\forall \{TP\}} m_B(TP)}{\sum_{\forall P \in \{FP\} \cup \{TP\}} m_B(P)} \leq \epsilon \tag{11}$$

- **Distance definition**:

$$D_{III}(N, \{P_i\}) = \min_i m_B(N, P_i) \tag{12}$$

Under Attack III with fuzzy matching, what a rational adversary would do is to bias the attack towards positive hospitals with a small non-matching score. Specifically, we consider the adversary pick a positive hospital with probability inversely proportional to the non-matching score.[7] In the previous example about a Georgia patient, the adversary would avoid choosing the New York hospital due to its high non-matching score (or long geographic distance).

Proposition: P^{3}I can mitigate Attack-III with assured ϵ-privacy in the sense of fuzzy-match semantic.

Proof: To the rational adversary, the success rate can be modeled by Equation 13.[8]

$$Pr(I^0|I, B) \quad = \quad \frac{\sum_{\forall TP \in I^0} m_B(TP)}{\sum_{\forall P \in I} m_B(P)} \tag{13}$$

The intuition of Equation 13 can be best illustrated by the Georgia patient example. Assuming there are five hospitals in the P^{3}I search result, and the distances of the five hospitals to the patient home are $2.5, 2, 1, .25, .2$ as in Table IV. Considering case 2 where there are two true positives, h_2 and h_3, the success rate following the calculation in Equation 13 is $Pr(I^0|I, B) = \frac{2+1}{2.5+2+1+.25+.2}$.

Plugging Equation 11 into Equation 13, we arrive at the ϵ-privacy: $Pr(I^0|I, B) \leq \epsilon$. ■

IV. Secure Distributed P^{3}I Construction with Efficiency

In the previous section, we assume a hypothetical authority unanimously trusted by all hospitals. In the real world, however, the assumption does not hold due to the mutual untrust among hospitals. In this section, we describe a distributed P^{3}I construction protocol without authority or mutual trust.

A. MPC-based Construction (M_0)

Given secure multi-party computation (MPC) supports generic computation, a straightforward approach is to map the entire construction-related computation (i.e. Algorithm 1) into MPC. This baseline approach is termed by M_0.

M_0 ensures data security due to MPC. The MPC that evaluates function top$k(I^0)$ in Algorithm 1 automatically

[7] Another strategy is for an adversary to deterministically pick the hospital with the maximal score, which ignores all score information other than the maximal one.

[8] The basic assumption is that all true-positive hospitals have a non-matching score close to zero.

guarantees the confidentiality of the sensitive input I^0. The output of P^{3}I construction is non-sensitive and can thus be made public.

However, this approach is inefficient and introduces prohibitively high overhead, especially in the HIE environment that deals with big health data and a large-scale network. Particularly, in the case of Algorithm 1, its circuit representation on which the MPC protocol operates is complex and deep due to the complex distance definition and stop condition evaluation. This leads to the computation overhead in MPC as demonstrated in our performance study of M_0 in § V-C2.

B. Construction by Pre-computation(M_1)

Our key observation to optimizing secure P^{3}I construction is that in function top$k(I^0)$ the private location metadata I^0, just one bit per party, is much smaller than public data which is the background knowledge that can relate to a variety of data sources and a large volume of data. To minimize the secure computation part, we perform aggressive *pre-computation*. The idea is to pre-compute on public background-knowledge data and all possible values of private data. Then when the private value arrives, we still use MPC to combine the private value with pre-computed results. Another benefit of pre-computation in P^{3}I is that the pre-computed data can be re-used multiple times for different patients, implying that the pre-computation only needs to run once.

The pre-computation based mapping is termed M_1. In M_1, the pre-computation evaluates top$k(\cdot)$ against all possible values in the domain of I^0. Given n hospitals (we consider a moderately large HIE network), the domain size of I^0 would be 2^n, and the pre-computed result is a mapping of 2^n entries, each one of which stores the result of top$k$ for a specific input value I^0. Given the pre-computed results, M_1 then securely looks up the entry corresponding to the private input $\underset{\sim}{I^0}$ (the underline indicates the private input), and finally outputs the pre-computed result on top$k(I^0)$. The secure look-up can be implemented by generic MPC or a multi-server PIR protocol, msPIR [31], [36].

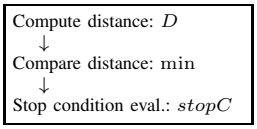
The security of M_1 comes from that the private input I^0 is fed into MPC without being disclosed to the pre-computation. The pre-computation only operates on the public data.

C. Partial Pre-computation (M_2, M_3)

Given M_1 pre-computes on the entire computation, our key idea is to partition the pre-computation to parts and map each part to either pre-computation or MPC.

The P^{3}I construction logic can be expressed by a pipeline of operations as in Figure 2. In this context, there can be naturally two partitioning points, resulting in different computation logics: 1) M_2 that partitions between phase B and C, and 2) M_3 that partitions between phase A and B.

In general, the two-stage computations of different partitioning approaches are illustrated in Equations 14 - 17. To start with, the baseline approach M_0 that puts everything into MPC is illustrated in Equation 14. In M_0, there is no pre-computation and the MPC stage performs the top-k computation described in Algorithm 1. Equation 15 describes the first optimization approach, M_1.



(a) Computation pipeline in P^{3}I construction

	PreComp	MPC+msPIR
M_0	-	D, min, $stopC$
M_1	D, min, $stopC$	PIR
M_2	D, min	PIR, $stopC$
M_3	D	PIR, min, $stopC$

(b) Partitioning

Fig. 2: P^{3}I construction model: The construction pipeline in Figure 2 consists of three steps: D) ComputeDistance, min) CompareDistance, and $stopC$) StopConditionEval. A basic idea to map the pipeline to our pre-computation framework is to partition at some point of the pipeline and map a partitioned pipeline to pre-computation. This gives rise to four possible partitioning/mapping approaches, M_0 - M_3 as in Figure 2b. While M_0 maps the entire pipeline into MPC, others ($M_1/M_2/M_3$) partition at some intermediate points (between $stopC$ and the end/min and $stopC$/D and min). The partitioned pre-computations are further illustrated in Equation 14 - Equation 17.

Equation 16 describes M_2. Here, the pre-computation needs to iterate through all possible values regarding I and j. For possible pairs $\forall\langle I, j\rangle$, it produces distances $\{D\}$ and stores them in $\mathcal{F}_y$. The complexity of the M_1's pre-computation and the size of pre-computed results are $\|\{I\}\| \times \|\{j\}\| = 2^n \cdot n$. In the MPC stage, it looks up the pre-computed distances by the private values $\langle I^0, j\rangle$ and produces the selected result.

Equation 17 describes M_3. The pre-computation is to compute all possible hospital-to-hospital distances (i.e. $Z_1(i,j), Z_2(i,j)$), thus resulting in the pre-computation complexity of $\|\{i\}\| \times \|\{j\}\| = n^2$. Then, in the MPC stage, it still performs most of the computation as needed, but can look-up, instead of computing, the hospital-to-hospital distance. The pre-computed result is a matrix with rows and columns representing hospitals.

D. Security Analysis

In § III-A, we analyze that the outcome of P^{3}I construction assures the ϵ-privacy under the considered attacks. This property still holds to the distributed P^{3}I construction (since the computation is virtually the same).

An extra source of privacy leakage is through the intermediate data produced during the process of distributed P^{3}I

Stage 1: Pre-computation $\mapsto$ Stage 2: MPC+msPIR

$$\overset{M_0}{\mapsto} \quad topk(\underset{\sim}{I^0})\{\min, D\} \tag{14}$$

$$topk(\{I^0\}) \quad \overset{M_1}{\mapsto} \quad \mathsf{msPIR}(\underset{\sim}{I^0}, topk(\{\cdot\})) \tag{15}$$

$$\mathcal{F}_y(\{\langle N, I^0\}) = \{D(N, I^0) | \forall N \notin I^0\} \quad \overset{M_2}{\mapsto} \quad topk(\underset{\sim}{I^0})\{\min, \mathsf{msPIR}(\langle \underset{\sim}{I^0}, j\rangle, \mathcal{F}_y)\} \tag{16}$$

$$\mathcal{F}_z(\{\langle N, P\rangle\}) = \{m_B(N, P) | \forall N \notin I^0, P \in I^0\} \quad \overset{M_3}{\mapsto} \quad topk(\underset{\sim}{I^0})\Big\{\min, D\{\min, \mathsf{msPIR}(\langle N, P\rangle, \mathcal{F}_z)\}\Big\} \tag{17}$$

construction. We call this type of privacy by the process privacy. Process privacy requires no private information leaked through the intermediate data. In M_1, the process privacy is assured by MPC since the entire computation is done in MPC. In the partial pre-computation (M_2 and M_3), the process privacy against a passive attacker (i.e. attacking by observing data instead of changing data) is assured by that 1) the pre-computation only takes non-private data and 2) all computations that depend on private data (e.g., I^0) are put inside MPC.

V. Experiments

To evaluate our solution in a holistic way, we conduct experiments to primarily answer the following two questions:

- How is the effectiveness of $\mathsf{P}^3\mathsf{I}$ in preserving patient privacy against background-knowledge attacks?
- What is the performance of $\mathsf{P}^3\mathsf{I}$ construction with a real-world big health dataset?

A. Dataset

USNEWS dataset: The USNEWS dataset [5] is used to model hospital profiles. The dataset considers 16 primary hospital-specialty categories, such as cardiology and rehabilitation (the entire list of specialties is shown in Table V). For each category, a hospital is associated with a rating of three grades: "Nationally ranked", "High-performing", and "Null". We map "Nationally ranked" to value 2, "High-performing" to value 1, and "Null" (i.e. the hospital does not have the department for this specialty) to value 0. Each hospital is associated with other profile information, such as the resident city and state. Currently, we select the dataset to include 40 top-ranked hospitals (out of 180) in the New York metropolitan area.

Open-NY Health Dataset ("Sparcs"): To model patient-wise hospital visits, we use an OPEN-NY dataset, called Sparcs [12]. The public dataset includes inpatient discharge records with identifiable information removed. At the finest granularity, it provides per-visit per-patient information (e.g., patient age group, gender, race, ethnicity and other de-identified information), the facility information (e.g., zip-code, name, service areas) and other per-visit information (e.g., admission type, the length of stay). Given the identifiable patient information is removed, we model the per-patient visit history by aggregating the records based on available identity information (i.e. age group, race, ethnicity, etc). In order to measure the attack accuracy, we manually tag the dataset by logically mapping each patient's visit to a specialty based on the description of medical procedure and diagnosis (those attributes are available in the Sparcs dataset). Then, we can measure how well the tagged specialty matches the actual hospital specialties, in a way to measure the attack success.

TABLE V: Specialty catalog in the USNEWS dataset

Index	Name	Index	Name
0	Cancer	8	Neurology & Neurosurgery
1	Cardiology & Heart Surgery	9	Ophthalmology
2	Diabetes & Endocrinology	10	Orthopedic
3	Ear, Nose & Throat	11	Psychiatry
4	Gastroenterology & GI Surgery	12	Pulmonology
5	Geriatrics	13	Rehabilitation
6	Gynecology	14	Rheumatology
7	Nephrology	15	Urology

B. Protection Effectiveness

In our security analysis, we consider a probabilistic attacker and the ϵ-privacy assurance, which come with two limitations: one is that it only considers attacks against one specific patient, and the other is that ϵ-privacy provides assurance in a statistical sense. To complement the security analysis, we move forward to measure the *variance* of success rate in a broader sense, that is, considering all patients.

Given the flexibility that the attacker now has in choosing which patient to attack, we consider the attacker can naturally exhaust all her options and target on the most vulnerable patient. The attacker can gauge the vulnerability of a patient by various metrics, such as, the one with the smallest number of specialties in her positive hospitals. Then, given the $\mathsf{P}^3\mathsf{I}$ is configured with a user-defined ϵ, we measure the actual success rates (of attacks on vulnerable patients) and report those that are larger than $epsilon$. In the experiment, we consider $epsilon = 0.4$ and 0.5. Note we deliberately avoid to use average success rate (by

multiple patients), since it is the largest success rate that makes the system vulnerable.

Overall effectiveness: We compare the case of P^{3}I with alternatives, including no-protection and grouping-based PPI. "No-protection" is the baseline which publishes raw location meta-data (i.e., patient-to-hospital information) without any noises. Grouping PPIs [15], [57] are based on the idea of K-anonymity (note to avoid confusion, we use K for K-anonymity, and k for top-k), which works by randomly grouping K hospitals together. We present our study results in Table VI. Here, for P^{3}I, we use the policy of adaptively choosing top-k to achieve a constant diversity l. To make the comparison fair, we use the same budget for injecting noises; that is, the amount of total false positives in P^{3}I is kept the same to that in grouping-based PPI. In the table, it is clear that P^{3}I achieves significantly smaller attack success rate and number of incidents.

TABLE VI: Effectiveness of P^{3}I

	Avg. success rate ($> .5$)	No. of incidents ($> .5$)
P^{3}I	0.537651	14
No-protection (Broadcasting)	0.782902	2349
PPIs [15], [57]	0.68231	1298

1) Effectiveness of top-k algorithm: We first report all incidents with success rates higher than the user-defined $\epsilon = 0.5$. The results are reported in Figure 3a where the x axis is the index of patients (in our processed health dataset, there are totally $280,000$ patients). It is easy to see that the no-protection approach results in much more densely distributed dots than P^{3}I under various configurations of k. Furthermore, it is often the case that no-protection results in 100% success rate, implying the real-world dataset is vulnerable to probabilistic attacks when without protection. This result is consistent to Table VI and explains the difference there.

We then manually vary the value of k to measure its effect on the attack success rate. The experiment result is presented in Figure 3b and 3c. It is interesting to see that it is not always the case that setting a larger k results in better protection; the protection in terms of larger-than-configured-ϵ incident rate is minimized at $k = 6$. Our preliminary inspection shows that this is relevant to the fact that real-world dataset is erroneous and does not fully match with some of our assumptions (e.g., patient does not always go to the nearest hospitals).

C. Performance Study of P^{3}I Construction

Given our two-stage MPC-optimizing framework, we measure the performance of both stages: pre-computation and MPC. After that, we present our performance study on multiple nodes across the Internet.

1) Pre-computation:

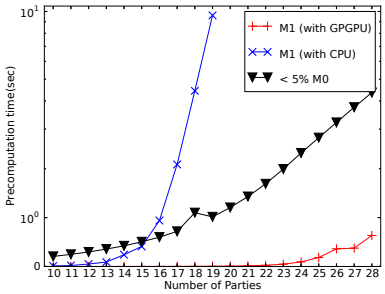


Fig. 4: Pre-computation performance

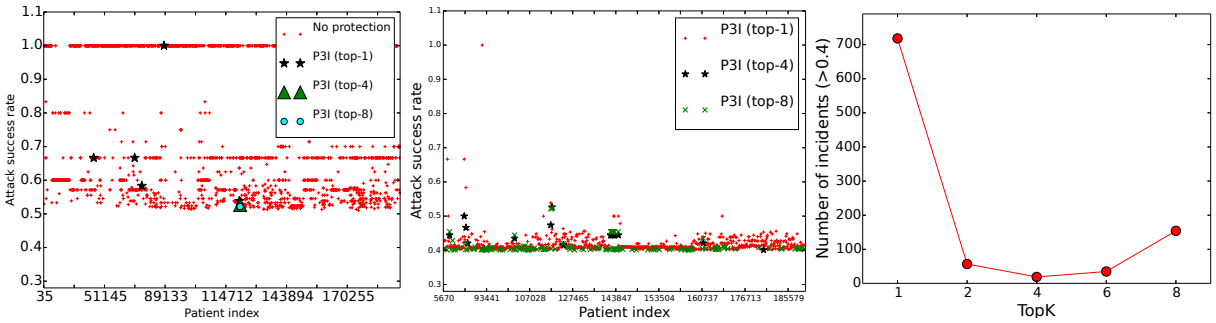
Implementation: Given the pre-computation considers possible combinations of input values, data-level parallelism can be exploited in the pre-computation by putting different possibilities into different threads without any inter-thread synchronization. We implement the multi-threaded pre-computation in two fashions, on CPU with pthread and GPGPU (general-purpose GPU) with CUDA.

Results: We compare the time to pre-compute on GPGPU and that on CPU with a standard time. We choose the standard time to be 5% of the time needed by M_0, because the pre-computation only makes sense when it is lightweight comparing to the MPC stage.

We show the performance results with up to 28 parties in Figure 4 where the CPU based pre-computation can scale up to 15 parties after which the CPU pre-computation time grows larger than our standard 5% of M_0 MPC time. The GPGPU based pre-computation has a relatively flat execution time which is significantly smaller than the standard time, implying that the GPGPU pre-computation is scalable at least with 28 parties.

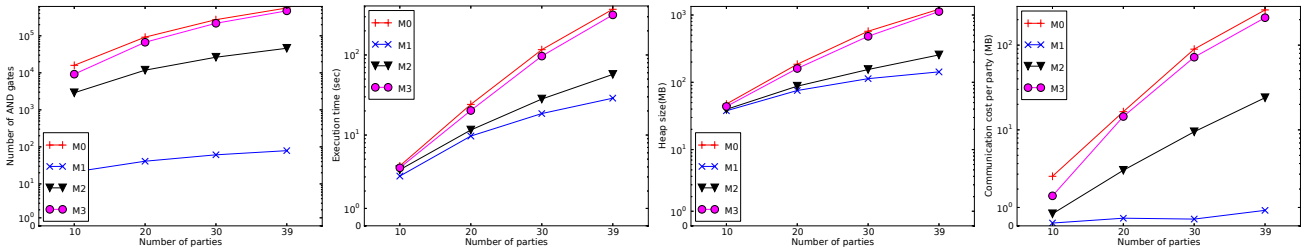
2) MPC Performance on Single-Node:

Implementation: The MPC stage in the P^{3}I construction is implemented using the GMW protocol [20], a standard open-source software in the applied cryptographic community for secure distributed computation. The GMW protocol exposes a low-level circuit-based programming interface that allows the programmer to write a circuit generator based on the intended computation logic. At runtime, the GMW protocol starts by each party generating the same circuit based on the program (i.e., the circuit generator). During the circuit evaluation, the protocol iterates through all the gates in the circuit (in the topologically sorted order) and for each gate. The evaluation of each gate is synchronized across all the parties, and its security is achieved by using two cryptographic primitives, that is, secret sharing [53] and oblivious transfer [48] applied at the granularity of individual single bits. The per-gate execution is to broadcast the shares of input-wire bit to all the parties in the entire network before locally evaluating the gate based on the received shares. Currently, we manually express the P^{3}I construction logic in the GMW boolean circuit, and tightly estimate the number of gates to pre-allocate so that the GMW performance



(a) Protection vs none (b) Comparing top-k (c) Number of incidents (> 0.4)

Fig. 3: Attack success rate



(a) Number of AND gates (b) Running time (c) Heap size per party (d) Comm. cost per party

Fig. 5: Performance of P^{3}I construction

sensitive to circuit size can be measured precisely. Our GMW-based implementation consists of about 1500 lines of C++ code and is open-sourced.[9]

TABLE VII: Experiment platform

New York Server	
CPU	Xeon(R) E5-2640 v3 @ 2.60GHz 2 processors/16 cores/32 hyper-threads
Memory	245 GB

California Server	
CPU	Xeon(R) E5-2687W @ 3.10GHz 2 processors/16 cores/32 hyper-threads
Memory	256 GB
GPGPU	Nvidia Tesla K20c 1 grid/65535 blocks/2^{27} threads Global Memory 5119MB

Execution platform: On the single-node setting, the performance study is conducted on a server machine with the hardware specification in Table VII. During the execution, up to 39 processes are launched, with each process representing a hospital. Each process holds a dedicated copy of the entire circuit allocated in its virtual memory space (i.e. without inter-process sharing). Given the large size of our memory (larger than 245 GB in total), the memory can hold the circuit copies of all 39 processes without too many paging activities.

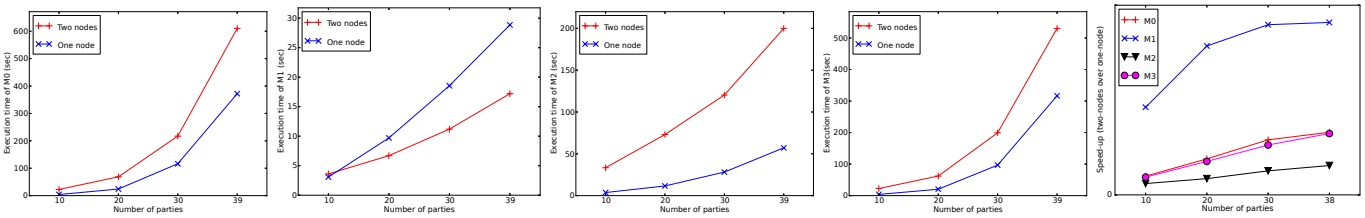
Results: To measure the performance, we mainly used four metrics, the number of AND gates in the compiled circuit, end-to-end execution time, memory consumption and communication costs.

- The GMW's Boolean circuit consists of XOR and AND gates. However, only AND gates are resource/time consuming because the XOR gates can be essentially evaluated locally for free (i.e. the free-XOR technique [21]). Thus, we only report the number of AND gates for performance study.
- We report the wall-clock time from the beginning of launching the first process to the end of the last process finishing its computation.
- We report the size of the heap memory in GMW that stores all circuit gates. This is measured by the Valgrind framework (particularly the Massif memory profiler [52]).
- We report the party-to-party communication overhead, by monitoring all outbound messages through the socket port of each process using IPTraf[10].

In the experiment, we vary the number of hospitals (or parties) and present the result in Figure 5. The results of the number of AND gates in Figure 5a and running time in Figure 5b are similar. They both show that the proposed optimization schemes (i.e. M_1, M_2, M_3) outperform the baseline approach without pre-computation, M_0. Most impressively, the M_1 approach performs the best with a speedup ratio of 13 times (comparing M_0) in running time with 39 parties. This is achieved due to the minimized

[9]https://github.com/sufullstacksecurity/phie

[10]http://iptraf.seul.org/



(a) M0 (b) M1 (c) M2 (d) M3 (e) Speedup

Fig. 6: Geo-distributed performance on the Internet

MPC computation by M_1 which offloads most of the computations (all four phases A, B, C and D) to the pre-computation stage and only conducts a table look-up operation inside MPC. In terms of memory consumption in Figure 5c, M_1 and M_2 are close, reducing up to 20% memory consumption comparing M_0 and M_3. Although the circuit of M_1 is much smaller than that of M_0, the memory reduction is offset by the relatively large pre-processed data of M_0. In Figure 5d, the communication overhead of M_1 stays to be the minimal among four approaches, achieving the saving of more than 2 orders of magnitude comparing the non-optimization approach M_0. This is consistent with the result in the number of AND gates.

3) MPC Performance on Multi-Nodes: We conduct the experiment with two servers set apart more than 2000 miles (one server in Syracuse, New York, and the other in San Diego, California). The bandwidth is 100 Mbps. The server specification is illustrated in Table VII. We measure the execution time of 4 approaches, and results are illustrated in Figure 6. For M_0, M_2 and M_3, the execution time on 2 nodes is longer than that on 1 node, implying that the performance slowdown by communication overhead outgrows the performance gain with extra hardware resources (e.g., CPU) on multiple nodes. M_1 is the only approach that improves the performance when deployed to more than one node, due to the minimized communication overhead and the performance likely being bounded by the CPU time.

VI. Related Work

A. Privacy-Preserving Data Federation

Multi-party noise generation: Distributed differential privacy [26], [60], [45] is proposed to support privacy-preserving aggregations. The randomized response [60] provides differential privacy yet with uncontrollable noises and loss of utility. PrivaDA [28] is proposed to achieve the optimal utility and performance optimization by adopting arithmetic circuit based MPC for the noise generation. Existing multi-party noise generation takes a randomized approach and mainly targets for statistical aggregation (e.g. distributed differential privacy). This is inapplicable to our problem which features deterministic noise generation for the rigorous privacy guarantee, and needs to serve non-aggregation queries.

PPI: Privacy-Preserving Index or PPI is proposed to federate and index distributed access-controlled documents [15], [14] and databases (e.g., patient medical records in the HIE locator service) [57] among autonomous providers. Being stored on an untrusted server, PPI entails preserving the content privacy of all participant providers or hospitals. Inspired by the privacy definition of K-anonymity [55], existing PPI work [15], [14], [57] follows the *grouping-based* approach; it organizes providers into disjoint privacy groups of size K, such that providers from the same group are indistinguishable. However, K-anonymity, while easy to construct, does not guarantee high-quality privacy preservation. In addition, early approaches of PPI construction are based on randomized responses [60], an iterative protocol that takes indefinite number of rounds to converge and may produce incorrect result (with certain probability). To avoid those drawbacks, ϵ-PPI combines randomized responses with a minimal use of multi-party computation to construct PPI correctly and efficiently.

Multi-party join: DJoin [44] is a federated database system built on top of multi-party joins, which are realized by privacy-preserving set intersections and general-purpose MPC for re-distributing noises. Its performance practicality has been demonstrated in small network with 3 to 5 parties. Multi-party joining has the potential to be applied in private record linkage problem (PRL) which is to match and link remote records of the same principle (e.g. patient in the healthcare domain) across multiple sites. While PRL has been studied for decades in the health-care domain, the recent advances include improved linking precision [34], providing privacy guarantee [19] and building a practical system [58], [6], [10]. Particularly in [19] the authors identify the performance problem of using MPC for PRL and propose to publish differential private synopsis of tables to avoid MPC and improve performance. Our work, focused on noising locator service, is orthogonal and complementary to the record linkage and joining, and can be integrated to an overall federated system of HIE.

B. Distributed Privacy-Preserving Mining

Distributed privacy-preserving data mining [59], [35] relies on algorithm/query-specific approaches to secure data-mining computations. For instance, association rule mining over vertically-partitioned databases [59], [35] reduces to scalar product which is secured by the impossibility of solving n equations in more than n unknowns. In addition, by assuming no collusion at all [18], [25], the secure data mining can be realized by efficient operations such as secret sharing and random number generation without using expensive protocols (e.g., oblivious transfers [48]). Our work is distinguished from privacy-preserving data mining in that we consider strong provable security against the worst-case collusion (e.g., all other parties may collude) which entails an extensive use of cryptographic protocols at fine granularity, rendering performance a critical issue.

C. MPC Frameworks and Optimization

In the last decade, practical MPC has attracted a large body of research work with a focus on programming language support and optimization [38], [18], [42], [32], [20], [17], [49], [16], [13]. Practical MPCs are built on top of cryptographic protocols, such as Yao's garbled circuits [61] or GMW protocol [30], with protocol-level optimization, such as Oblivious Transfer (OT) extensions [33], or for stronger security, such as resilience with dishonest majority [23]. The MPC protocols assume a circuit interface to express the computation, and practical programming support focuses on compiling a program written in a high-level language into the circuit. Existing MPC protocols and systems mainly focus on a small-scale computing that involves 2 or 3 parties. To the general MPC problem, a fundamental trade-off exists between performance and computation generality; for instance, randomized responses [60] and other techniques for privacy-preserving data mining take an ad-hoc and domain-specific approach, which can be efficient at scale. By contrast, the general-purpose MPC is rather expensive.

MPC Optimization: High performance overhead stays to be one of the major hurdles to applying MPC in practice, which is partly caused by MPC's fine-grained use (e.g., per single bit) of expensive cryptographic primitives, and the need to transfer all *possible* computation results for the "obliviousness" of computation flow. Various optimization techniques are proposed to utilize the programming semantics to reduce the circuit size and depth (e.g., by using the hardware synthesis tools [54], [24]) and optimize the resource utilization (e.g., just-in-time compilation and pipelined execution [32], [38]). Program analysis [37] is used to automatically infer privacy-sensitive data and constraints MPC only to the sensitive data. Our MPC optimization is currently specific to the $\mathsf{P}^3\mathsf{I}$ construction problem, while holding the potential to apply to more generic computations.

D. Anonymization Definitions

Publishing public-use data about individuals without revealing sensitive information has received a lot of research attentions in the last decade. Various anonymization definitions have been proposed and gained popularity, including K-anonymity [55], l-diversity [41], t-closeness [40], and differential privacy [27]. In addition, prior work [43] formally studied the information leakage under background knowledge attacks by formulating the problem using a proposed declarative language. These anonymity notions however are generally inapplicable to the PPI problem – they are mainly designed for statistic analysis or aggregation style computation where the result is global per-table data, while PPI needs to serve queries specific to individual records.

r-confidentiality [63] is a privacy notion specific to the PPI problem. It assumes a probabilistic attacker on PPI and considers the increase of attack success-rate with/without using the background knowledge. By contrast, our proposed ϵ-privacy considers to bound the attack success-rate (instead of the increase) which we believe provides better privacy control.

VII. Conclusion

In summary, this paper presents the $\mathsf{P}^3\mathsf{I}$ framework, a privacy-preserving indexing and searching for the Health Information Exchange. The key design of $\mathsf{P}^3\mathsf{I}$ is to address the conflict between the assured privacy preservation (resilient to background-knowledge attacks) and efficiency of secure distributed construction. The proposed solution is a two-stage optimization framework which separates the computation on the sensitive data and non-sensitive background knowledge. The pre-computation is efficiently realized by exploiting data-level parallelism on GPGPU. We implement the proposed technique on real open-source software, and through extensive experiment study, demonstrate the performance improvement by more than one orders of magnitudes.

References

[1] Commonwell rls: http://www.commonwellalliance.org/services.

[2] Gahin: http://www.gahin.org/.

[3] Healtheconnections: http://www.healtheconnections.org/rhio.

[4] Hipaa, http://www.cms.hhs.gov/hipaageninfo/.

[5] http://health.usnews.com/best-hospitals/area/new-york-ny/specialty.

[6] Nextgate: http://www.nextgate.com/our-products/empi/.

[7] Nhin connect, http://www.connectopensource.org/.

[8] Nhin: http://www.hhs.gov/healthit/healthnetwork.

[9] Ohio voter files: http://www2.sos.state.oh.us/pls/voter/f?p=111:1.

[10] Openempi: http://www.openempi.org/.

[11] Shin-ny: http://www.nyehealth.org.

[12] Sparcs: http://www.health.ny.gov/statistics/sparcs/.

[13] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner. More efficient oblivious transfer and extensions for faster secure computation. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 535–548, 2013.

[14] M. Bawa, R. J. Bayardo, Jr, R. Agrawal, and J. Vaidya. Privacy-preserving indexing of documents on the network. *The VLDB Journal*, 18(4), 2009.

[15] M. Bawa, R. J. B. Jr., and R. Agrawal. Privacy-preserving indexing of documents on the network. In *VLDB*, pages 922–933, 2003.

[16] M. Bellare, V. T. Hoang, S. Keelveedhi, and P. Rogaway. Efficient garbling from a fixed-key blockcipher. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 478–492, 2013.

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

[18] D. Bogdanov, S. Laur, and J. Willemson. Sharemind: A framework for fast privacy-preserving computations. In *Computer Security - ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, October 6-8, 2008. Proceedings*, pages 192–206, 2008.

[19] J. Cao, F. Rao, E. Bertino, and M. Kantarcioglu. A hybrid private record linkage scheme: Separating differentially private synopses from matching records. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 1011–1022, 2015.

[20] S. G. Choi, K. Hwang, J. Katz, T. Malkin, and D. Rubenstein. Secure multi-party computation of boolean circuits with applications to privacy in on-line marketplaces. In *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*, pages 416–432, 2012.

[21] S. G. Choi, J. Katz, R. Kumaresan, and H.-S. Zhou. On the security of the free-xor technique. In *Theory of Cryptography*, pages 39–53. Springer, 2012.

[22] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 280–299, 2001.

[23] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, pages 1–18, 2013.

[24] D. Demmler, G. Dessouky, F. Koushanfar, A. Sadeghi, T. Schneider, and S. Zeitouni. Automated synthesis of optimized circuits for secure computation. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 1504–1517, 2015.

[25] W. Du and M. J. Atallah. Protocols for secure remote database access with approximate matching. In *E-Commerce Security and Privacy*, pages 87–111. 2001.

[26] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, pages 486–503, 2006.

[27] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[28] F. Eigner, M. Maffei, I. Pryvalov, F. Pampaloni, and A. Kate. Differentially private data aggregation with optimal utility. In *Proceedings of the 30th Annual Computer Security Applications Conference, ACSAC 2014, New Orleans, LA, USA, December 8-12, 2014*, pages 316–325, 2014.

[29] N. Ferguson, B. Schneier, and T. Kohno. *Cryptography Engineering - Design Principles and Practical Applications*. Wiley, 2010.

[30] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229, 1987.

[31] R. Henry, F. G. Olumofin, and I. Goldberg. Practical PIR for electronic commerce. In *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011*, pages 677–690, 2011.

[32] Y. Huang, D. Evans, J. Katz, and L. Malka. Faster secure two-party computation using garbled circuits. In *USENIX Security Symposium*, 2011.

[33] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, pages 145–161, 2003.

[34] P. Jurczyk, J. J. Lu, L. Xiong, J. D. Cragan, and A. Correa. Fril: A tool for comparative record linkage. *AMIA annual symposium proceedings*, 2008:440, 2008.

[35] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Trans. Knowl. Data Eng.*, 16(9):1026–1037, 2004.

[36] M. Keller and P. Scholl. Efficient, oblivious data structures for MPC. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 506–525, 2014.

[37] F. Kerschbaum. Automatically optimizing secure computation. In *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011*, pages 703–714, 2011.

[38] B. Kreuter, A. Shelat, B. Mood, and K. R. B. Butler. PCF: A portable circuit format for scalable two-party secure computation. In *Proceedings of the 22th USENIX Security Symposium, Washington, DC, USA, August 14-16, 2013*, pages 321–336, 2013.

[39] L. Lamport. Password authentification with insecure communication. *Commun. ACM*, 24(11):770–772, 1981.

[40] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 106–115, 2007.

[41] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *ICDE*, page 24, 2006.

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

[43] D. J. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Y. Halpern. Worst-case background knowledge for privacy-preserving data publishing. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 126–135, 2007.

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

[45] M. Pettai and P. Laud. Combining differential privacy and secure multiparty computation. In *Proceedings of the 31st Annual Computer Security Applications Conference, Los Angeles, CA, USA, December 7-11, 2015*, pages 421–430, 2015.

[46] R. A. Popa, F. H. Li, and N. Zeldovich. An ideal-security protocol for order-preserving encoding. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 463–477, 2013.

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

[48] M. O. Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptology ePrint Archive*, 2005:187, 2005.

[49] A. Rastogi, M. A. Hammer, and M. Hicks. Wysteria: A programming language for generic, mixed-mode multiparty computations. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, pages 655–670, 2014.

[50] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.

[51] B. Schneier. Metadata = surveillance. *IEEE Security & Privacy*, 12(2):84, 2014.

[52] J. Seward, N. Nethercote, and J. Weidendorfer. *Valgrind 3.3-Advanced Debugging and Profiling for GNU/Linux applications*. Network Theory Ltd., 2008.

[53] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[54] E. M. Songhori, S. U. Hussain, A. Sadeghi, T. Schneider, and F. Koushanfar. Tinygarble: Highly compressed and scalable sequential garbled circuits. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 411–428, 2015.

[55] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[56] Y. Tang, L. Liu, A. Iyengar, K. Lee, and Q. Zhang. e-ppi: Locator service in information networks with personalized privacy preservation. In *IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014, Madrid, Spain, June 30 - July 3, 2014*, pages 186–197, 2014.

[57] Y. Tang, T. Wang, and L. Liu. Privacy preserving indexing for ehealth information networks. In *CIKM*, pages 905–914, 2011.

[58] C. Toth, E. Durham, M. Kantarcioglu, Y. Xue, and B. Malin. Soempi: A secure open enterprise master patient index software toolkit for private record linkage. In *AMIA Annual Symposium Proceedings*, volume 2014, page 1105. American Medical Informatics Association, 2014.

[59] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 639–644, 2002.

[60] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[61] A. C. Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167, 1986.

[62] A. C.-C. Yao. Protocols for secure computations (extended abstract). In *FOCS*, pages 160–164, 1982.

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