Secure and Efficient Multi-Party Directory Publication for Privacy-Preserving Data Sharing

Katchaguy Areekijseree Yuzhe Tang Ju Chen
Shuang Wang † Arun Iyengar ‡ Balaji Palanisamy♯

Department of EECS, Syracuse University, NY, USA,
{kareekij,ytang100,jchen133}@syr.edu
†*Department of Biomedical Informatics (DBMI),UCSD, CA, USA*
shw070@ucsd.edu
‡*IBM T.J. Watson Research Center, Yorktown Heights, NY, USA,*
aruni@us.ibm.edu
♯*School of Computing and Information, University of Pittsburgh, Pittsburgh, PA, USA*
bpalan@pitt.edu

Abstract. In the era of big-data, personal data is produced, collected and consumed at different sites. A public directory connects data producers and consumers over the Internet and should be constructed securely given the privacy-sensitive nature of personal data.
This work tackles the research problem of distributed, privacy-preserving directory publication, with strong security and practical efficiency. For proven security, we follow the protocols of secure multi-party computations (MPC). For efficiency, we propose a pre-computation framework that minimizes the private computation and conducts aggressive pre-computation on public data. Several pre-computation policies are proposed with varying degrees of aggressiveness. For systems-level efficiency, the pre-computation is implemented with data parallelism on general-purpose graphics processing units (GPGPU).We apply the proposed scheme to real health-care scenarios for constructing patient-locator services in emerging Health Information Exchange (or HIE) networks.
We conduct extensive performance studies on real datasets and with an implementation based on open-source MPC software. With experiments on local and geo-distributed settings, our performance results show that the proposed pre-computation achieves a speedup of more than an order of magnitude without security loss.

1 Introduction

In the era of big-data, personal data is produced, collected and consumed in digital forms, bringing unprecedented convenience to the society. As data production and consumption are decoupled at different sites, sharing person-specific data over the Internet becomes a popular application paradigm as widely observed in a variety of domains ranging from electronic healthcare, social networks, Internet of things, malware detection, to many others.

A public directory service is a crucial data-sharing component. In a data-sharing workflow, a data consumer queries the directory service to locate the producer sites that may have the documents of interest. The directory service maintains the private producer-location information, and connects data consumers and producers. For instance, in electronic healthcare, HIE or Healthcare Information Exchange is an emerging data-sharing platform [10, 4] where the directory called locator service [1, 9, 11, 5] helps a doctor (data consumer) find the electronic medical records (EMR) of a patient (data producer). The data-location information ("which hospitals a patient has visited")

may reveal privacy-sensitive facts; for instance, knowing that a celebrity visited a rehabilitation center, one can infer that s/he may have a drug problem.

A naive way of constructing the directory is for any data producer to directly publish its list of associated people (e.g., the list of patients having visited a hospital). However, this approach discloses the private data-location information to network adversaries performing traffic analysis. This privacy disclosure leaks "identifiable information" and would violate data-protection laws (e.g., HIPAA in USA [6], EC95-46 in European Union [3] and various privacy laws in Asian countries [56]) that govern the data-sharing across borders in regulatory domains.

This work tackles the problem of distributed and privacy-preserving publication of directory, with strong security and high efficiency. In our problem, data producers are operated autonomously and they distrust each other. The publication problem can be modeled as a secure Multi-Party Computation (MPC) problem [73, 37, 25, 52, 27] where a joint computation with inputs private to different parties is evaluated in a proven-secure fashion. A naive instantiation of the directory publication is by embedding entire publication logic in an MPC protocol, which however causes high overhead and is impractical, because of the expensive cryptographic primitives used in constructing an MPC. A conventional remedy is to identify the private part of the computation (e.g., by data-flow analysis [59, 17]) and to map only this part to the MPC. Unfortunately, this approach is not effective in our problem, as the private and public data flows of the directory-construction logic are inter-tangled and separating them becomes difficult.

In this work, we propose an aggressive pre-computation technique that minimizes (instead of separating) the private computation for multi-party directory publication. Concretely, we conduct the pre-computation by considering all possible values of private data. It then applies expensive MPCs to a simple selection logic, that is, select from the list of pre-computed results by the actual value of private data. At the first glimpse, this optimization technique may seem counter-intuitive as the pre-computation augments the input space exponentially. In practice, particular to our directory construction problem, its effectiveness relies on the application characteristic: The public computation is usually bulky and private identity data is much smaller. For instance, achieving the privacy of t-closeness [51] entails complex computation on the public background knowledge, such as similarity/distance calculation. With a global identity management system, the private identifiable data is minimal. In addition, we propose several policies that vary in the degree of pre-computation aggressiveness. The policies can help the optimization technique adapt to concrete scenarios with different private-data sizes.

To improve the system efficiency, we leverage the data-level parallelism and implement the pre-computation on General-Purpose Graphics Processing Units (GPGPU). We implement our design on real MPC software [25] and conduct performance evaluation in both local and geo-distributed settings. Our evaluation verifies the pre-computation speedup by more than an order of magnitude over the conventional approach. Through evaluation on real-world datasets, the assurance of privacy preservation is also verified.

The contributions of this work are listed as following:

- We address the research of constructing privacy-preserving directory in emerging data-sharing applications. We model the general problem as a distributed privacy-preserving data publication problem.
- We propose an application-specific techniques for MPC pre-computation in the directory publication. The insight is based on that the public background knowledge in privacy-preserving publication can be isolated from expensive MPC. We implement this optimization design on real MPC software.
- We propose systems-level optimization by data-parallel pre-computation. We implement the optimization on GPGPU.

– We conduct performance evaluation and demonstrate an order of magnitude performance speedup.

The rest of the paper is organized as following: § 2 formulates the research problem. The proposed technique, pre-computation based MPC for directory publication, is presented in § 3. A case study in healthcare domain is described in § 4. Performance evaluation is presented next in § 5. We discuss the generalizability and extensions of the proposed technique in § 6. § 7 surveys the related work and § 8 concludes the paper.

2 Research Formulation

This section presents the system and threat model, the security goals, survey of existing techniques, and preliminary on privacy-preserving data publication algorithms.

2.1 System Model

The target eco-system involves three roles: data producers, data consumers, and the host of directory service. Each data producer owns a table of personal records where each record is keyed by the identity of the owner of this record. Given a person of interest, a data consumer would want to find his/her records at all producer sites. The directory service helps the consumer "discover" relevant data producers who maintain the result records.

Formally, sharing personal records in our system works in two steps: First, a data consumer interested in a person's records poses a query to the directory service and looks up the list of producers who have this person's records. Then, the consumer contacts individual producers and locally searches the records there. In this process, the query is based on a personal identity, which we assume is known globally. In practice, this global identity can be maintained physically by an identity-management server or constructed virtually such as by patient record linkage in healthcare [70, 43].

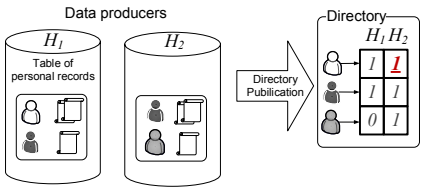


Fig. 1: System model of public directory: Two data producers share three people's records. In the directory, value one means presence and zero means absence (e.g., producer H_1 does not have gray person's records). The underscored one in red is a false positive in the sense that producer H_2 does not have the record of the white person but the directory records the opposite (for the sake of privacy preservation).

We assume each data producer locally has a data-protection mechanism in place (e.g., user authentication and authorization) that prevents an external party from accessing the records without data owner's consent. Figure 1 illustrates the abstract model of our system. The model is applicable to data-sharing applications in regulatory domains; A concrete example is about sharing patient electronic medical records (EMR)

in healthcare information exchange networks, where data producers are hospitals, personal data are patients' EMRs and consumers can be physicians diagnosing patient. The details of the scenario will be elaborated in § 4.

The target computation of this work is about building the directory. A baseline is that each data producer sends its local access-control list to the third-party directory which enforces the access control when serving the directory requests. This baseline however becomes problematic when the directory host is untrustworthy (e.g., by third-party clouds): First, enforcing access control with integrity entails user authentication and authorization to be done by a trusted party. Second, the local access-control list reveals the binding between a person and her data producers, which can be privacy-sensitive in many applications. For instance, in Healthcare scenarios, the binding between a patient and a rehabilitation center can reveal that this person may have a drug problem. Even when the directory is protected by the host, an adversary can easily recover the binding by performing network traffic analysis and extracting this information from the side-channel of the consumer access trace.

We consider the privacy-preserving publication of directory. Existing data-privacy definitions, such as k-anonymity, l-diversity, t-closeness, are applicable to our problem. For instance, k-anonymity requires k people have their published lists of producers to be the same. l-diversity requires people placed in the same group have l distinct lists of producers. t-closeness requires each group of people to have similar producer lists to all the producers. In this work, we mainly use the notion of ϵ-privacy [69] to drive further presentation. The main idea of ϵ privacy is to bound the amount of noises or false positives in the published list of producers by a percentage of ϵ. We will discuss the application of our technique to other privacy definitions in § 6.2.

Formally, the notion of ϵ-privacy is adapted to our threat model by being aware of background knowledge — we make the false positive producers indistinguishable from true positives, such that the distribution of true positives is similar to that of false positives. What's noteworthy is that the similarity is measured on the dimension of external, public knowledge. For instance, in HIE, the similarity between hospitals (producers) can be defined by hospital specialties and geographic locations.

Privacy-preserving publication algorithm: Achieving ϵ-privacy can be done by a top-K algorithm. Concretely, given a list of true positive producers, the algorithm finds K negative producers which are closest to the positive ones. The value of K can be simply calculated from ϵ and the number of true producers $|T|$. Listing 2 presents the top-K algorithm which entails the iterative computation of nearest neighbor with similarity/distance defined by public knowledge. The distance computation depends on the metric that represents producers and it can be Euclid distance, Hamming distance, and others.

2.2 Threat Model and Security Goals

This work targets on the *distributed publication of privacy-preserving directory* with untrusting data producers. In our problem, a data producer runs autonomously and distrusts external parties including peer producers. Data producers get engaged in the distributed computation for publishing privacy-preserving directory where they exchange information with each other.

In the threat model, an adversary can eavesdrop all messages being exchanged during the distributed directory publication. For a producer, the adversary can be a network eavesdropper or a peer producer. Formally, this is the semi-honest model used in formulating a secure multi-party computation problem [21], where the adversary, being a participant in the computation, honestly follows the protocol execution but is curious about any data that flows through her during the execution. Multiple adversaries may collude. Given a network of n producers, we consider the collusion can be up to $n - 1$ peer producers.

```
TopK(true_producers T, all_producers S){
  R = T;
  while (less than K iterations){
    //find NN in S to T
    for (any j in S){
      for (any i in T){
        min_dist_j = min(dist(T[i],S[j]),min_dist_j);
        min_j = j;
      }
      min_dist = min(min_dist,d);
    }
    R.add(S[min_j]);
    S.remove(S[min_j]);
  }
  return R;
}
```

Fig. 2: Top-K algorithm to achieve privacy

The security goal is to assure the data security in the directory-publication process. Our security goal is to ensure perfect privacy (in an information-theoretic sense). Informally, it means an adversary's view only depends on her input and public output. In other words, the messages exchanged in the protocol execution when the input of other parties take one value are "indistinguishable" from those when the input of other parties take another value. More formal treatment of the MPC data security can be found on classic texts [27].

Our threat model and security goal fit in the real-world requirement for policy compliance in data sharing. In many regulatory domains, a data producer has the responsibility of protecting the personal data it maintains and complying data-protection laws. For instance, HIPAA [6] states any identifiable information about a patient cannot be shared to any third-party, without the patient's consent.

Non-goals of this work include directory data authenticity, producer-site data protection, key management, etc. Encrypting data on the directory is orthogonal, as the content of directory is anyway disclosed to the adversary of network eavesdropper performing traffic analysis.

2.3 Preliminary on Multi-Party Computation

In our protocol, we make use of existing multi-party computation (MPC) protocols whose background is presented here. In general, the purpose of MPC is to evaluate a function whose inputs are provided by different parties. Each input is private to its provider party. The protocol of MPC ensures that it does not leak any information about the private inputs even when the computation states are exchanged and shared. Different computational models exist in MPC, including circuit and RAM. After decades of studies, there are a variety of MPC protocols realizing different computation models, specialized for different network scales (for two, three or many parties). In particular, the protocol of GMW [37] is for multi-party, Boolean-circuit based MPC that is constructed based on the primitives of secret-sharing and oblivious transfers. The protocol of multi-server Private-Information Retrieval (ms-PIR) [39, 46] is a RAM-based MPC with multiple servers interacting a client on the computation of a simple selection operation (e.g., like a database selection).

MPC causes high overhead, mainly due to the "data -oblivious" representation of the computation and cryptographic primitives being used in the construction. For more-

than-three party computation, the use of secret sharing also cause high overhead as the shares need to be broadcast in the entire network. This unscalability (in data and network sizes) makes it challenging to apply MPC for real-world distributed applications.

In practice, the common way MPC is used for many-party distributed applications is based on the "outsourcing" paradigm. That is, given multiple input parties, the GMW protocol distributes the input shares to a small number of computing parties (e.g., three parties as in the Sharemind system [22]). The data security heavily relies on the non-collusion assumption of the computing parties. In our work, we deem this outsourcing model unsuitable for the target application. In HIPAA, a hospital cannot share patient data with *any* third-party entity without patient consent. Therefore, our problem considers each input party as computing party and the MPC protocol needs to run directly on a medium or large network.

3 Secure Directory Publication with Pre-Computation

In this section, we present the secure directory publication and the optimization techniques based on pre-computation. The general idea is to abstract the computation at different levels and precompute the computation at a specific level. This way, we present a series of precomputation techniques (in § 3.2, § 3.3) that vary in their aggressiveness. To start with, we present the naive approach based on multi-party computation (MPC) without precomputation.

3.1 MPC-based Publication

Privacy-preserving directory publication is an MPC problem as the input data are spread across multiple producers and are private to them. The naive way to realize directory publication is thus to place the computation as in List 2 into the MPC; this approach is denoted by M_0. Given the circuit representation of MPC program, the algorithm in List 2 can be easily converted to a circuit; the algorithm is a nested loop with pair-wise distance computation, and the data/control flow is essentially oblivious. In particular, we represent each producer by a vector (e.g., specialties of a hospital) and the similarity between producers can be realized by hamming distance. More complex string similarity computation is realized by dynamic-programming based algorithms which are also data oblivious. The security of this approach inherent from that of MPC.

This MPC approach is inefficient especially in big-data sharing scenario where there are a large amount of personal records. This is due to the expensive cryptographic primitives (e.g., oblivious transfers, etc) used in MPC protocols. To improve the performance, it relies on reducing the use of MPC in the distributed directory publication.

3.2 Full Precomputation Scheme

To reduce the use of MPC, we propose application-level precomputation. Given the $topK(T, S)$ algorithm in List 2 where only input T (the true producers) is private, we pre-compute the algorithm on the public input S and all possible values of private input T. The precomputation result is a table of results under different T values. Then, we use the actual value of T to privately look up this table and to securely retrieve the result entry. This stage can be realized in MPC using protocols such as multi-server private information retrieval (ms-PIR) [39, 46]. Formally, the full precomputation is to compute $topK(2^S, S)$ where 2^S is the power set of S which includes all possible values of private T. This scheme is named M_1.

The precomputation is effective in our directory-publication problem, provided the following characteristics. First the $topK$ algorithm invokes some complex computation

such as distance computation (i.e. Line 7 in List 2) which involve background knowledge about the producer profiles (e.g., hospital specialties and geographic locations). Precomputation avoids placing these complex computations in MPC which reduces overhead. Second, the precomputation only needs to be done once and its results can be *reused* for publishing different people's entries. Third, given the independence between different values, one can leverage data-parallelism to facilitate the computation. Note that the precomputation needs to be done for all possible value of T, that is, the power set of all producers; although the possibility combination grows exponentially with the number of producers, we only consider the data-producer network is moderately large. For instance, in healthcare, a regional or statewide HIE typically consists of less than hundreds of hospitals in a consortium.

The security of precomputation relies on the fact that no private value is involves in the precomputation. Private data only occurs in the actual MPC computation.

3.3 Selective Precomputation Schemes

The full precomputation scheme considers the directory computation of $topK$ as a whole for precomputation. In this section, we dive into the computation $topK()$ and *selectively* precompute certain computation-intensive parts in $topK()$. Concretely, our selective technique considers $topK$ consists of distance-computation at different granularity. For one, it is to pre-compute the distance between T and $S - T$, considering all possible values of T. This way, we have the selective precomputation, M_2. For the other, it is to pre-compute the distance between all pairwise data producers. This yield the selective precomputation scheme, M_3.

In M_2, the precomputation considers all possible values of true-producer T. Given a value T^*, it precomputes the set-wise distance between T^* and $S - T^*$. This produces a distance table for the subsequent MPC. In the MPC, it first follows the computation in List 2 until Line 6. Then for Line 6 to 9, it is replaced by a secure lookup into the precomputation table. The lookup is realized by the ms-PIR protocol as in M_1.

In M_3, it precomputes the pair-wise distance matrix. That is, for any producer s_1 and $s_2 \in S$, it precomputes their distance and stores it in a table. Then, in the MPC stage, it follows the algorithm in List 2 except that the call to `dist(T[i],S[j])` is replaced by a ms-PIR lookup to the precomputation table.

The security of these precomputation schemes are straightforward, as all private-data related computations are placed inside the MPC/ms-PIR protocol whose security is proven. The precomputation only considers the public data.

In summary, the $topK$ computation for privacy-preserving directory publication can be modeled as a process that issues a series of call to `dist(T[i],S[j])`. Our pre-computation schemes partitions this computation process at different "break" points and selectively places a certain partition to precomputation and the rest of computation into MPC/ms-PIR. Table 1 illustrates the three pre-computation schemes from this computation-partitioning perspective.

3.4 Data-Parallel Pre-Computation

The pre-computation handles multiple independent input values. There is innate data parallelism that can be exploited for better performing pre-computation. In our system, we realize it by data-parallel pre-computation tasks where each task with distinct input value runs in a dedicated thread. Different threads run concurrently and without synchronization. We implement this data-parallel pre-computation framework on both multi-core CPU and general-purpose GPU (GPGPU). Given the large number of possibilities in input values (and the simplicity of each task), GPGPU lends itself to the parallel pre-computation due to its scalable execution model.

Table 1: Partitioning $topK$ algorithm to the precomputation-MPC framework: For notation in this table, T, S are true and all producers as in the $topK()$ algorithm in List 2. D_i for $i = 1, 2, 3$ are the table storing precomputation results. MPC is secure multi-party computation protocol and $msPIR$ is a special MPC protocol for multi-server private information retrieval.

	Pre-compute	MPC+msPIR
M_0	-	$topK(T, S)$
M_1	$D_1 = topK(2^S, S)$	$Lookup_{msPIR}(D_1, T)$
M_2	$D_2 = dist(2^S, S)$	$topK2_{MPC}(T, S)$ invoking $Lookup_{msPIR}(D_2, T)$
M_3	$D_3 = dist(S, S)$	$topK3_{MPC}(T, S)$ invoking $Lookup_{msPIR}(D_3, T[i], S[j])$

In implementation, the CPU implementation is based on pthread library [13]. We pack multiple possible input values in one thread and the number of threads is twice the number of hyper-threads in hardware. The GPGPU implementation is based on CUDA library [2]. In this case, the underlying NVidia-Tesla GPU has global memory of 5 GB and threads run in one grid of 65,635 blocks, each of 1024 GPU threads. This architecture allows to scale the number of threads to 2^{27} and can easily handle the producer networks of more than 27 parties.

4 Case Study: Healthcare Locator

In this section, we present the case study of applying our public locator service in healthcare information exchange networks (HIE). HIE is a health data-sharing network where the data is patient electronic medical records (EMR), data producers are hospitals where each patient visit results in the generation of new entries in an EMR, and data consumers are clinical doctors. A typical application scenario is effective sharing patient's EMR during a clinical visit where the doctor diagnosing a patient needs to view the relevant EMRs of the patient which are produced and stored in remote hospitals.

In this setting, our threat model and security goal apply. Patient EMRs are personal, privacy-sensitive documents, the sharing of which must comply HIPAA [6]. Each hospital has its local information-security infrastructure in place (e.g., access control and user authentication).

A directory service, called HIE locator, can be used to facilitate the EMR sharing between hospitals and to help discovery of a patient's previous hospitals. In the normal case, the list of hospitals is discovered by the doctor asking for it to the patient. However, this is error prone (e.g., the patient forgets about it) and is inapplicable in emergency (e.g., the patient is sent to hospital unconscious). Our privacy-preserving directory can complement the common workflow to improve the quality of healthcare.

Figure 3 illustrates the abstract workflow of sharing EMRs in HIE networks. In a clinic scenario, Alice, the patient, is seeing a physician (data consumer) who interacts with HIE network (directory) to locate the hospitals Alice visited before (data producer). In real HIE applications, the locator service runs healthcare software (e.g., OpenEMPI [11]) and is hosted by Amazon AWS alike public clouds. The public clouds are not trustworthy and it entails the use of our privacy-preserving directory protocol for publishing the HIE locator. Concretely, the life cycle of an EMR, including the data-sharing flow, can be divided into three stages: 1) EMR production where Alice's EMRs are generated or updated to reflect her clinical visit; here we assume Alice has given consent on delegating the EMR to the "producer" hospitals. 2) Locator (periodical) publication where the EMR updates are published to the public directory of HIE

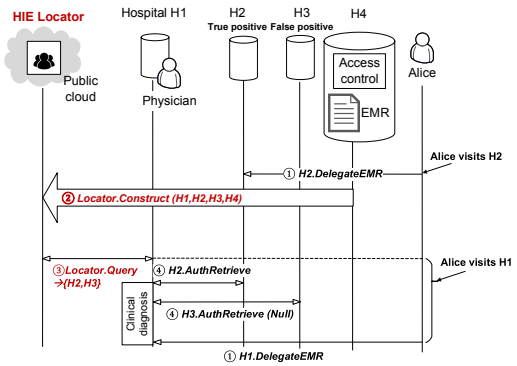


Fig. 3: Data-sharing workflow in the HIE: The figure illustrates how Alice's medical record (EMR) stored on Hospital H_2 is used. It shows the entire life cycle of this EMR: The EMR was produced when Alice paid a clinical visit to H_2 (1). During the current visit in Hospital H_1, Alice's physician requires accessing her EMR in H_2 (3). The physician first contacts the third-party locator service hosted on a public cloud (which is constructed at an earlier time (2)) and obtains the list of candidate hospitals H_2 and H_3. Here, H_3 is a noise for privacy preservation purpose. The physician then contacts both H_2 and H_3, and find EMR on H_2 (4). Note that the physician can do so because she has the credential to access data on both H_2 and H_3. For an adversary obtaining the list of H_2 and H_3, she cannot distinguish which hospitals are noise as she does not have the credential.

locator in a privacy-preserving fashion. This is when our directory publication protocol is being invoked in the overall HIE workflow. 3) Locator service where the locator serves the physician's request to locate Alice's producer hospitals (3.1) and find the EMRs of interest there (3.2). In particular for stage 3.2, after the physician obtains the list of potential hospitals (including both true and false positive ones), he will contact each hospital and find EMRs by going through the local user authentication and access control there.

Security analysis: In this data-sharing process, the EMR is produced and stored securely (stage 1)) by assuming the producer hospitals' secure and trustworthy local healthcare infrastructure (e.g., faithfully enforcing access control and honest health IT administrator).

The security in publishing the healthcare locator (stage 2)) is based on the security of our privacy-preserving directory protocol, which is further based on the security of MPC protocols [21] and computation-partitioning schemes.

The security in serving the healthcare locator (stage 3.1) is based on the fact that sufficient amount of noise has been injected into the directory, such that an Internet adversary performing traffic analysis and knowing the list of hospitals contacted by the physician can not distinguish between the true positive hospitals and noises. The formal notion of indistinguishability is presented in related work [69, 54, 68].

The security in searching and retrieving records on individual hospitals (stage 3.2)) is ensured by the security of local healthcare IT (for enforcing access control) and the secure channel on the Internet (e.g., https and underlying PKI [35]).

5 Evaluation

In this section, we study the feasibility of our technique for HIE applications in a holistic manner. Lacking benchmark dataset in existing literature, we first present a real

healthcare dataset to populate the HIE data producers and locator. This sets up a target scenario for the performance study which we will present next. The purpose of performance evaluation is to answer the following question: *What is the overhead of privacy-preserving directory publication? and how effective is the proposed precomputation technique in performance optimization?*

5.1 Dataset

USNEWS dataset The USNEWS dataset [7] 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 2). 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 [14]. 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 quasi-identity information (i.e. age group, race, ethnicity, etc).

Table 2: Specialty catalog in the USNEWS dataset

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

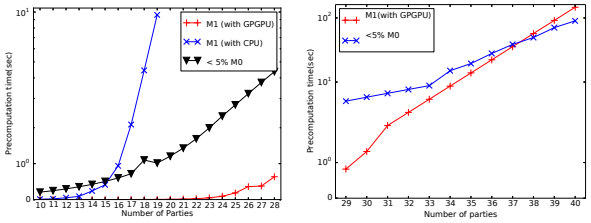
Table 3: Experiment platform

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

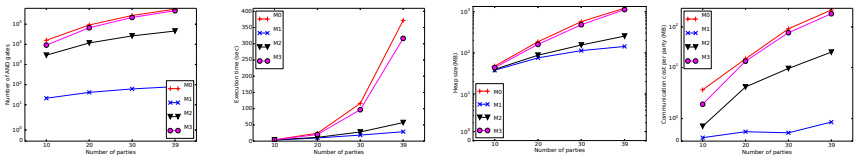
5.2 Performance of Directory Publication

We first conduct micro-benchmark to test the performance of data-parallel precomputation. Then, we test the overall performance of secure directory publication, with a machine of multi-core processor and in a geo-distributed setting.



(a) With ≤ 28 parties (b) With > 28 parties

Fig. 4: Pre-computation performance with GPGPU



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

Fig. 5: Performance of directory publication based on precomputation and MPC

Micro-benchmark of Pre-computation The pre-computation is implemented with data parallelism (as described in 3.4) and runs on multi-core CPU and GPGPU. We report the time to pre-compute on GPGPU and that on CPU in Figure 4. This figure also includes a baseline which is the 5% execution time of running M_0 (i.e. without any precomputation).

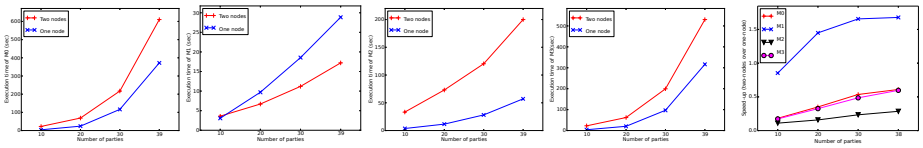
The performance result in Figure 4a shows that GPGPU based pre-computation is effective in reducing the execution time, and its overhead is negligible comparing the baseline. Concretely, the CPU based precomputation has its execution time to quickly surpass the baseline when the network grows over 15 parties. The GPGPU-based pre-computation has much lower overhead than the baseline for any network with less than 28 parties.

For more than 28 parties, all GPU threads are occupied and it will need multiple iterations in transferring data from GPU's global memory to host memory. As a result, the GPGPU precomputation time increases exponentially, also reported in the Figure 4b (note that the y axis is of log scale). With a single GPGPU card, the precomputation time surpasses the baseline when the network is larger than about 40 parties. Here, we stress that the typical scale of a healthcare consortium is usually medium-sized (e.g., tens of hospitals and clinical centers). For nation-wide healthcare systems, there may be thousands of hospitals. In this case, one can use more GPGPU cards to do the precomputation in parallel, while retaining the efficiency.

Overall Performance with MPC The MPC-based implementation of directory publication is realized on the GMW software [25], an open-source MPC software and Percy++ [12], an open-source multi-server PIR software. We note that our precomputation protocol only relies on the general MPC and PIR interface and other MPC "backend" software can be used in our protocol. The GMW protocol exposes a circuit-based programming interface that requires MPC programmers to write a generator for Boolean circuit encapsulating the intended computation logic. At runtime, the GMW protocol runs on multiple parties where each party generates and executes the circuit by iterating through all gates in the circuit (following a topologically sorted order); for each gate, the evaluation is synchronized across all parties. The GMW protocol makes bit-wise use of two cryptographic primitives which provides the security of the protocol, that is, secret sharing [65] and oblivious transfer [62]. In particular, the per-gate evaluation in GMW is to broadcast the shares of input-wire bit to all the parties in the entire network. In our application, we manually express the logic of $topK$ algorithm in the GMW Boolean circuit, and tightly estimate the number of gates to pre-allocate so that the unused GMW circuit can be optimized out. Our GMW-based implementation consists of about 1500 lines of C++ code.

Multi-processing execution platform: We first run our protocol on a single node with multi-processing. The machine specs are in Table 3 (the New York server). In this setting, each process represents a data producer and runs a GMW party. In the execution, each process holds a dedicated copy of the entire circuit allocated in its virtual-memory space and without shared memory. The machine has memory large enough (245 GB in total) to hold all circuit copies of the 39 parties without paging.

Results on multi-processing: To measure the performance of MPC, we used four metrics, the number of AND gates (1), end-to-end execution time (2), memory consumption (3) and communication costs (4). 1) We report the number of AND gates in the compiled GMW Boolean circuit. This metric helps evaluate the performance in a hardware-independent fashion. We only consider AND gates in a circuit and ignore other gates (i.e., XOR gates) because evaluating XOR is free (i.e. free-XOR technique [26]) and evaluating AND gates dominates the cost. 2) We report the wall-clock time from launching the first process to the completion of the last process. 3) We report the size of the heap memory in GMW that stores all circuit gates. It is measured by the



(a) Baseline M_0 (b) Precomputation M_1 (c) Precomputation M_2 (d) Precomputation M_3 (e) Speedup

Fig. 6: Geo-distributed performance on the Internet

Valgrind framework (particularly the Massif memory profiler [64]). 4) We report the party-to-party communication overhead, by monitoring all outbound messages through the socket port of each process using IPTraf[1].

In the experiment, we vary the number of parties (or data producers) and present the result in Figure 5. Figure 5a reports the result of AND gate number and Figure 5b reports wall-clock running time. They both show that the pre-computation based schemes (i.e. M_1, M_2, M_3) outperform the baseline without pre-computation. Notably, the M_1 scheme causes the best performance with a speedup of 13 times (comparing the baseline M_0) in the setting of 39 parties. This result demonstrates the effectiveness of pre-computation techniques that off-loads computation from the expensive MPC. In terms of memory consumption in Figure 5c, M_1 and M_2 are close, reducing up to memory consumption roughly by an order of magnitude comparing M_0 and M_3. It shows that while M_1 produces pre-computation results as additional data, its much smaller circuit (for simple lookup operation in ms-PIR) makes the overall saving of memory footprint as compared to the baseline M_0. In Figure 5d, the communication overhead of M_1 stays to be the smallest among the four schemes, with a saving of more than 2 orders of magnitudes comparing M_0. This is consistent with the result in the number of AND gates.

Geo-distributed execution platform: We conduct the experiment with two servers set apart more than 3000 miles (one server in the State of New York, and the other in the State of California). The bandwidth is 100 Mbps. The specification of the two servers is illustrated in Table 3. Each server runs half of the parties with multiprocessing. Different parties communicate through sockets. The precomputation runs only in one server.

Results with geo-distributed execution report the execution time of the four schemes in the geo-distributed setting. The results are in Figure 6. For comparison, we include the results in the single-node setting. The execution time grows super-linearly with the number of parties in a network. For M_0, M_2 and M_3, running them on two geo-distributed nodes leads to longer execution time. Interestingly for M_1, the geo-distributed execution is faster than the single-node one. In this case, the performance slowdown caused by the slower communication channels is offset by the performance gain from the extra hardware (e.g., CPU) on multiple nodes. We suspect this performance result is due to that the MPC is dominated more by the local computations (on secret shares) and less by the network communications.

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

6 Discussion

In this section, we consider the generalizability and extensions of the proposed technique beyond Healthcare Locators. We present the extension to new computation beyond exact-match lookup as in HIE locator and new privacy definitions beyond ϵ privacy.

6.1 Similarity-based Directory

In this application scenario, a data consumer may want to find data about a group of "similar" people. Comparing the HIE locator that performs exact-match lookup (by a person's ID), the directory here performs similarity search. Take the healthcare domain as an example. The similarity-based directory (e.g., PatientLikeMe.com) entails publishing the binding between a patient, say Alice, and the hospitals that store the EMRs of patients who are similar to Alice, where the similarity can be defined on their syndrome, genome and other bio-medical features.

This directory can be constructed by extending the $topK$ algorithm in Listing 2. In particular, the list of true producers is interpreted as the producers storing the data of people who are similar to the person being queried. The person-person similarity is defined on the external background knowledge as mentioned above. In Line 7 of Listing 2, the producer-producer distance is defined on the external domain knowledge (e.g., hospital profiles in specialty, locations, etc).

6.2 Achieving Other Privacy Definitions

Our proposed technique can also be naturally extended to achieving other privacy definitions including k-anonymity, l-diversity, and t-closeness. A general framework for these privacy definitions is that the original data is a table of sensitive data key, quasi-identifier and public attributes. Achieving a specific privacy definition entails finding a "partitioning" solution that partitions the data-table rows based on quasi-identifiers into groups such that each group in the result will meet the privacy definition. There are different algorithms such as generalization, suppression, perturbation, etc. In this section, we consider the Mondrian algorithm [50] that represents the data records as a point in a multi-dimensional space (assuming multi-dimensional quasi-identifiers) and partitions the space by following kd-tree schemes and greedily refining the partition to the smallest units.

One can express the Mondrian algorithm naturally in MPC and the pre-computation stays effective because of extensive complex computation (e.g., computing distances) are based on public knowledge.

In details, adapting the Mondrian algorithm to directory publication can be done by following: For each person the producer vector is her multi-dimensional quasi-identifier and her identity is the data attribute that needs to be searched publicly. In the Mondrian algorithm, it entails determining whether it allows to split the current partition (1) and if it does, finding the dimension and split value (2). In (1) and (2) it involves complex distance computation; for instance, l-diversity requires counting the number of distinct producer vectors in a partition and t-closeness requires computing the group-wise similarity (among all records in a group). The number of distinct producer-vectors and group-wise similarity can both be realized by a nested loop where each iteration computes the pair-wise similarity between two records.

6.3 Data Updates

In many applications, data is being continuously generated. In this case, directory is constructed and updated in batches. In each batch, the latest updates are reflected in the

directory incrementally. The $topK$ algorithm can be evaluated for every batch by interpreting the list of `all_producers` to all negative producers before the updates and the list of `true_producers` to be the new producers in the current batch. The content stored in the directory can be time-series data that each update batch is materialized independently.

7 Related Work

7.1 Privacy-Preserving Data Federation

Multi-party noise generation Distributed differential privacy [32, 72, 61] is proposed to support privacy-preserving aggregations. The randomized response [72] provides differential privacy yet with uncontrollable noises and loss of utility. PrivaDA [34] 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 [19, 18] and databases (e.g., patient medical records in the HIE locator service) [69] 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 [67], existing PPI work [19, 18, 69] 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 [68, 54] are based on randomized responses [72], 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 [60] 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 [44], providing privacy guarantee [24] and building a practical system [70, 8, 11]. Particularly in [24] 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.

7.2 Distributed Privacy-Preserving Mining

Distributed privacy-preserving data mining [71, 45] relies on algorithm/query-specific approaches to secure data-mining computations. For instance, association rule mining

over vertically-partitioned databases [71, 45] 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 [22, 31], 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 [62]). 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.

7.3 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 [48, 22, 57, 40, 25, 21, 63, 20, 16]. Practical MPCs are built on top of cryptographic protocols, such as Yao's garbled circuits [73] or GMW protocol [37], with protocol-level optimization, such as Oblivious Transfer (OT) extensions [41], or for stronger security, such as resilience with dishonest majority [28]. 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 [72] 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 [66, 29]) and optimize the resource utilization (e.g., just-in-time compilation and pipelined execution [40, 48]). Program analysis [47] is used to automatically infer privacy-sensitive data and constraints MPC only to the sensitive data. [49] conducts pre-processing on verification of MPC and results in general transformation from a passively secure protocol to an actively secure one. Our MPC optimization is currently specific to the directory construction problem, while holding the potential to apply to more generic computations.

Some programming frameworks support high-level programming languages with compilers (e.g., Fairplay(MP) for SFDL [57, 21], Sharemind for SecreC [42], CBMC-GC for ANSI C [36], PCF for C [48], Wysteria for a high-level typed specification language [63], PICCO for C with extension [75]), while others expose a quite low-level circuit based interface (e.g., GMW [25], JustGarble [20], OTExtension [16]); particularly both boolean circuit (e.g., GMW) and arithmetic circuit (e.g., SEPIA [23]) are considered. In addition, some advanced technique designs based on hybrid model that combines both boolean or arithmetic circuits (e.g., ABY [30], TASTY [38], Wysteria [63]).

7.4 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 [67], l-diversity [55], t-closeness [51], and differential privacy [33]. In addition, prior

work [58] 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 [74] 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.

8 Conclusion

This work presents an MPC-precomputation framework tailored for privacy-preserving data publication for data-sharing applications. The pre-computation framework improves the performance by minimizing the private-data computation and realizing the public-data only pre-computation in a data-parallel fashion. Several pre-computation policies are proposed with varying degrees on the aggressiveness. It is demonstrated that the proposed pre-computation scheme is applicable in real health-care scenarios. Based on real datasets and implementation on open-source MPC software, the performance study shows that the proposed pre-computation achieves a speedup of more than an order of magnitude without security loss.

Acknowledgement

The authors would thank anonymous reviewers for their constructive suggestions. The first three authors were supported by the Cyber Research Institute in Rome, NY, under Grant Number #28254. Shuang Wang was supported by NIH R00HG008175.

References

1. CommonWell: http://www.commonwellalliance.org/.
2. Cuda: https://en.wikipedia.org/wiki/cuda.
3. Directive 95/46/ec of the european parliament and of the council.
4. GaHIN: http://www.gahin.org/.
5. HealthEConnections: http://www.healtheconnections.org/rhio.
6. HiPAA, http://www.cms.hhs.gov/hipaageninfo/.
7. http://health.usnews.com/best-hospitals/area/new-york-ny/specialty.
8. Nextgate: http://www.nextgate.com/our-products/empi/.
9. NHIN Connect, http://www.connectopensource.org/.
10. NHIN: https://www.healthit.gov.
11. OpenEMPI: http://www.openempi.org/.
12. Percy++/pir in c++: http://percy.sourceforge.net/.
13. pthread: https://en.wikipedia.org/wiki/posix_threads.
14. SPARCS: http://www.health.ny.gov/statistics/sparcs/.
15. *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*. IEEE Computer Society, 2015.
16. 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.
17. J. Bater, G. Elliott, C. Eggen, S. Goel, A. N. Kho, and J. Duggan. SMCQL: secure query processing for private data networks. *CoRR*, abs/1606.06808, 2016.

18. 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.
19. M. Bawa, R. J. B. Jr., and R. Agrawal. Privacy-preserving indexing of documents on the network. In *VLDB*, pages 922–933, 2003.
20. 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. IEEE Computer Society, 2013.
21. A. Ben-David, N. Nisan, and B. Pinkas. Fairplaymp: a system for secure multi-party computation. In P. Ning, P. F. Syverson, and S. Jha, editors, *ACM Conference on Computer and Communications Security*, pages 257–266. ACM, 2008.
22. 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.
23. M. Burkhart, M. Strasser, D. Many, and X. A. Dimitropoulos. SEPIA: privacy-preserving aggregation of multi-domain network events and statistics. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 223–240. USENIX Association, 2010.
24. 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.
25. 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 O. Dunkelman, editor, *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*, volume 7178 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2012.
26. 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.
27. R. Cramer, I. Damgård, and J. B. Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.
28. 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 J. Crampton, S. Jajodia, and K. Mayes, editors, *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, volume 8134 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2013.
29. D. Demmler, G. Dessouky, F. Koushanfar, A. Sadeghi, T. Schneider, and S. Zeitouni. Automated synthesis of optimized circuits for secure computation. In I. Ray, N. Li, and C. Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 1504–1517. ACM, 2015.
30. D. Demmler, T. Schneider, and M. Zohner. Aby - a framework for efficient mixed-protocol secure two-party computation. In *Network and Distributed System Security Symposium (NDSS'15)*, Feb. 2015.
31. 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.
32. C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In S. Vaudenay, editor, *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*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006.
33. 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.
34. 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.
35. N. Ferguson, B. Schneier, and T. Kohno. *Cryptography Engineering - Design Principles and Practical Applications*. Wiley, 2010.

36. M. Franz, A. Holzer, S. Katzenbeisser, C. Schallhart, and H. Veith. CBMC-GC: an ANSI C compiler for secure two-party computations. In A. Cohen, editor, *Compiler Construction - 23rd International Conference, CC 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8409 of *Lecture Notes in Computer Science*, pages 244–249. Springer, 2014.
37. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In A. V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987.
38. W. Henecka, S. Kögl, A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Tasty: tool for automating secure two-party computations. In *ACM CCS*, pages 451–462, 2010.
39. 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.
40. Y. Huang, D. Evans, J. Katz, and L. Malka. Faster secure two-party computation using garbled circuits. In *USENIX Security Symposium*. USENIX Association, 2011.
41. 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.
42. R. Jagomägis. Secrec: a privacy-aware programming language with applications in data mining.
43. P. Jurczyk, J. J. Lu, L. Xiong, J. D. Cragan, and A. Correa. FRIL: A tool for comparative record linkage. In *AMIA 2008, American Medical Informatics Association Annual Symposium, Washington, DC, USA, November 8-12, 2008*, 2008.
44. 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.
45. 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.
46. 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.
47. 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.
48. 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.
49. P. Laud and A. Pankova. Preprocessing-based verification of multiparty protocols with honest majority. *IACR Cryptology ePrint Archive*, 2015:674, 2015.
50. K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In Liu et al. [53], page 25.
51. 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.
52. C. Liu, X. S. Wang, K. Nayak, Y. Huang, and E. Shi. Oblivm: A programming framework for secure computation. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015* [15], pages 359–376.
53. L. Liu, A. Reuter, K. Whang, and J. Zhang, editors. *Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, 3-8 April 2006, Atlanta, GA, USA*. IEEE Computer Society, 2006.
54. Y. T. L. Liu. Privacy-preserving multi-keyword search in information networks. *IEEE Trans. Knowl. Data Eng.*, 27(9):2424–2437, 2015.
55. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In Liu et al. [53], page 24.
56. A. B. Makulilo. *Asian Data Privacy Laws, Trade and Human Rights Perspective*, by graham greenleaf. *I. J. Law and Information Technology*, 23(3):322–324, 2015.
57. D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella. Fairplay - secure two-party computation system. In M. Blaze, editor, *USENIX Security Symposium*, pages 287–302. USENIX, 2004.

58. 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.
59. S. McCamant and M. D. Ernst. Quantitative information flow as network flow capacity. In *Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation, Tucson, AZ, USA, June 7-13, 2008*, pages 193–205, 2008.
60. A. Narayan and A. Haeberlen. DJoin: differentially private join queries over distributed databases. In *OSDI*, Oct. 2012.
61. 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.
62. M. O. Rabin. How to exchange secrets with oblivious transfer. *IACR Cryptology ePrint Archive*, 2005:187, 2005.
63. 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. IEEE Computer Society, 2014.
64. J. Seward, N. Nethercote, and J. Weidendorfer. *Valgrind 3.3-Advanced Debugging and Profiling for GNU/Linux applications*. Network Theory Ltd., 2008.
65. A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
66. 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* [15], pages 411–428.
67. L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
68. 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.
69. Y. Tang, T. Wang, and L. Liu. Privacy preserving indexing for ehealth information networks. In *CIKM*, pages 905–914, 2011.
70. 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.
71. 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.
72. S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.
73. 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. IEEE Computer Society, 1986.
74. 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.
75. Y. Zhang, A. Steele, and M. Blanton. PICCO: a general-purpose compiler for private distributed computation. In *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 813–826, 2013.