Privacy-Preserving Computation over Genetic Data: HLA Matching and so on

Jinming Cui , Huaping Li , Meng Yang
BGI, Shenzhen, Guangdong, China
{cuijinming, lihuaping1, yangmeng1}@genomics.cn

Abstract

Genetic data is an indispensable part of big data, promoting the advancement of life science and biomedicine. Yet, highly private genetic data also brings concerns about privacy risks in data sharing. In our work, we adopt the cryptographic primitive *Secure Function Evaluation (SFE)* to address this problem. A secure SFE scheme allows institutions and hospitals to compute a function while preserving the privacy of their input data, and each participant knows nothing but their own input and the final result. In our work, we present privacy-preserving solutions for *Human Leukocyte Antigen (HLA)* matching and two popular biostatistics tests: *Chi-squared test* and *odds ratio test*. We also show that our protocols are compatible with multiple databases simultaneously and could feasibly handle larger-scale data up to genome-wide level. This approach may serve as a new way to jointly analyze distributed and restricted genetic data among institutions and hospitals. Meanwhile, it can potentially be extended to other genetic analysis algorithms, allowing individuals to analyze their own genomes without endangering data privacy.

1 Introduction

Gene is the unit of heredity made of DNA, transferred from a parent to his/her offspring, and the complete set of genes and other genetic materials present in a cell or organism generate genome. With the progress of life science, genetic data has been proved to be closely related to most of the life activities, such as growth and illnesses [Shendure *et al.*, 2019]. The popularity of genetic testing leads to an exponential growth of biological data, while this large amount of genetic data provides valuable insights in fields like healthcare, drug discovery and precision medicine [Liu *et al.*, 2018] [Lee *et al.*, 2018]. On the other hand, individual's blood relationship, disease susceptibility, physical characteristics and personality traits — all can be interpreted from genes. Concerns over genetic data privacy impede the communication of data and prevent joint analysis among different institutions. Meanwhile, countries enacted laws or regulations to strictly control the usage and sharing of genetic data. Thus, most of the genetic data is kept in overwhelmingly secured repositories to provide high-level data security and privacy.

In fact, human genetic information is diversified. Biology studies like statistical analysis [Vento-Tormo *et al.*, 2018], data selection, and data matching will greatly benefit from more comprehensive and massive data resources. Currently, institutions/hospitals rely on signed agreements to share private genetic data under the supervision of governments, but the signing process is rather slow, and private data will be permanently shared. National authorities have established a centralized public database to alleviate those problems. For example, ClinGen, founded in 2013 by National Institutes of Health (NIH), aims to become a central system that defines the clinical relevance of genes and variants in precision medicine and research. However, data-centralization also worsens the consequence of the security breach of the database, which potentially attracts more attacks. Furthermore, centralized database usually suffers from proof accessibility. In this data-driven era, to promote the development of life science, isolated data is far from enough.

In this paper, we utilize the cryptographic primitive related to secure computation called *Secure Function Evaluation*, present a privacy-preserving *Human Leukocyte Antigen (HLA) matching* solution and an extensible secure statistics computation framework. In summary, we make the following contributions:

- We design the first secure HLA matching solution, adopting the secure function evaluation protocol based on the boolean circuit. Also, we propose a database shuffling mechanism to increase the security of HLA matching.
- We provide an extensible framework for developing other statistics algorithms, which can take common bioinformatics files as input. We also discuss the security problems of naive implementation of secure statistics function and provide possible solutions.

2 Related Work

2.1 Genome Privacy

The biomedical community has been working to utilize the benefit of data sharing. However, the integration of genetic data yields many security and privacy concerns. On the one hand, genetic data carries various sensitive information about

an individual such as ethnic heritage, disease predispositions, and phenotypic traits. On the other hand, due to its hereditary nature, the disclosure of genetic data would bring an impact on their blood relatives. Also, the recently adopted EU General Data Protection Regulation (GDPR) refers to the term genetic data throughout its recitals, and — perhaps more importantly — genetic data are included in the list of sensitive data in Article 9, without any kind of differentiation. It indicates that the legislator tries to reflect the increasing use of genetic data (not only for research purposes) and the growing concerns around such use.

Over the years, privacy researchers have proposed several methods on data de-identification and cryptographic privacy-preserving techniques [Gymrek *et al.*, 2013] [Homer *et al.*, 2008] [S. Shringarpure and Bustamante, 2015], meanwhile other researchers try to find ways to breach genetic privacy. [Erlich and Narayanan, 2013] provides a good summary for current techniques either related to breach or protect genetic privacy. Some of those attacks have even been demonstrated in practice [Homer *et al.*, 2008].

In recognition of the privacy needs for genetic data, governments and institutions across the world have launched several projects and software with the aim to promote data sharing in a privacy-preserving manner, e.g. *the Beacon Project*, *the Matchmaker Exchange*.

2.2 Secure Computation

Secure Computation, also called *Secure Function Evaluation (SFE)* is a cryptographic primitive which securely evaluates a function depending on two or more private inputs. Informally, SFE allows two parties, Alice and Bob, to secure compute a function $f(x, y)$, where x is Alice's private input and y is Bob's private input, SFE ensures that at the end of the protocol both parties could know nothing but their own input and the evaluation result $f(x, y)$.

The first generic circuit-based SFE protocol "Garbled Circuit" was proposed by [C. Yao, 1982] in 1982. In the late 1990s several protocols have been proposed based on different circuits, e.g. boolean circuit, arithmetic circuit [Goldreich *et al.*, 1987] [Ben-Or *et al.*, 1988] [Beaver *et al.*, 1990]. Apart from the work we mentioned above, researchers have been designing protocols for particular problems sacrificing flexibility for efficiency such as *private set intersection (PSI)*. Later in section 4, we discuss our choices between different circuits and how we choose our PSI protocol.

There are also other cryptographic primitives allowing the secure computation of a function, for example, *fully homomorphic encryption (FHE) scheme*. However, FHE scheme by now is far from practical and requires enormous computation power, therefore in our work, we have not considered it as one of the solutions.

3 Preliminaries

3.1 Notation

In our HLA matching scenario, we denote the server as S and the client as C. An HLA allele (string format) is denoted as x. An individual HLA data $I^{|l|} = \{x_1, x_2, ..., x_l\}$ consists of l HLA allele. The database owned by S with n entries is denoted as $D^{|n|} = \{I_1^{|l_1|}, ..., I_n^{|l_n|}\}$. We further encode HLA allele from string format into 32-bit integer format, the encoded data is denoted as $[x]$. As for secure statistics computation protocol, we consider it in a two-party setting. P_1 and P_2 are two parties holding database D_1 and D_2 separately. In addition, the underlying cryptographic primitives of our protocols are instantiated with security parameter by $\kappa = \{80, 112, 128\}$ and static security parameter by $\sigma = 40$, as is recommended by National Institude of Standards and Techonology (NIST).

3.2 Data Format

Human Leukocyte Antigens (HLA) are proteins found on most cells in human body and are used to match patients and donors for bone marrow or cord blood transplants. Each person has multiple HLA alleles and to deal with the encoding format of the highly polymorphic HLA molecules, WHO Nomenclature Committee for Factors of the HLA System [HLA.Alleles.org, 2010] has proposed the nomenclature of HLA alleles. Our HLA matching solution follows their HLA system nomenclature and takes standard HLA alleles as input. An example of HLA allele name is given below:

*HLA-A*02:101N*

By definition, each HLA allele name has a unique number corresponding to 2 sets of digits separated by colons. The last character describes the protein expression level. For instance, character N indicates that this HLA gene does not express, therefore we remove HLA data with 'N' during data pre-processing. HLA typing bases on real DNA sequence or polypeptide chain (the translated product from DNA sequence). As alleles receive at least a 4-digit number, any missing specific HLA protein will be replaced by *01*. We converted string format HLA gene to an integer, the conversion rule is shown in table A.

In the matching case, we work with the HLA database consisting of HLA allele information and sample IDs. As is shown above, *HLA-A* is the HLA allele type, *02* stands for the allele group and *101* stands for the specific HLA protein.

For secure statistics computation protocol, we use HapMap format files as input, which is a popular format adopted by several major international biological databases describing genotype information of different samples. The current release consists of sample attributes, including SNP ID, SNP allele and list of sample names. The following Table 1 shows an example of HapMap format file.

rs#	alleles	NA18532	NA18605	NA18542
rs11252546	C/T	TT	CC	CC
rs10904494	A/G	AA	AG	GG
rs11591988	A/C	AA	AA	CC
...	...	...	...	...

Table 1: HapMap format

3.3 Security Model

In this paper, we assume that each participating pair has already established secure point-to-point channels prior to the

execution of protocol, in other words, all P2P communications are assumed to be securely encrypted.

Normally, secure function evaluation protocols are discussed under two security models: *semi-honest security* (also called *honest-but-curious security*) and *malicious security*. A semi-honest adversary is believed to execute protocol honestly, but he/she will try to learn as much information as possible from the message it receives. In contrast, the malicious security model assumes that adversaries could perform arbitrary behavior. Our secure protocols are based on a semi-honest secure computation protocol, therefore in our work, we choose the semi-honest model.

4 Privacy-Preserving HLA Matching

HLA molecules are highly polymorphic and they play an important role in the immune system. HLA matching is the main way for both organ and bone marrow transplantation patients to find a suitable donor. The accuracy and efficiency of matching strongly affect the postoperative rehabilitation and survival rate of patients, a higher degree of HLA compatibility of donor-patient means lower incidence of transplantation rejection, which increases the success rate of transplantation. There may be a higher HLA matching rate between immediate siblings or parents. However, if matching between siblings or parents failed, comparing with donor database seems to be a more promising way. By now, there is still a lack of comprehensive and unified HLA database and hospitals have their own HLA database with trivial overlap. Currently, each patient may need to take multiple HLA matching tests at different hospitals to find a potential donor. And each of these tests costs approximately three thousand yuan, which is a heavy financial burden for patients' families. In the conventional offline HLA matching, only 3 to 6 HLA classical genes will be considered within a limited donor group. With our privacy-preserving HLA Matching system, patients would only need to take the HLA matching test once and can utilize all HLA allele information.

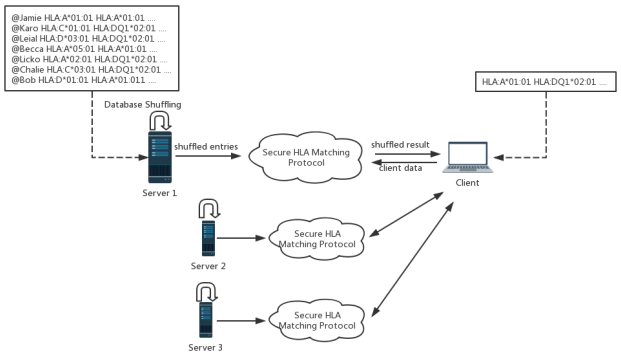

Figure 1: Privacy-Preserving HLA Matching Scenario

In our solution, we bring our HLA matching system online, enabling efficient and convenient privacy-preserving matching with different databases. From the perspective of patients, they can easily access to more diverse and massive donor resources, increasing the possibility of successful matching. As for databases, our solution ensures data security and privacy under a semi-honest model. Figure 1 illustrates the structure of our solution.

In this section, we discuss and detail our privacy-preserving HLA matching solution. To begin with, a client (presumably a patient) wants to securely find out the similarity between his/her HLA data and each entry in the server's database. We let server S holds an HLA database $D^{|n|} = \{I_1^{|l_1|}, ..., I_n^{|l_n|}\}$, and client C holds one individual HLA data $I_c^{|l_c|}$. Clinically, the similarity between two individuals' HLA data could be evaluated by calculating the number of the same HLA alleles, denoted as $f(I_1, I_2)$. This similarity function could also be treated as a slightly modified version of generic Private Set Intersection protocol $f(I_1, I_2)$, which takes two input sets I_1 and I_2, returning the size of their intersection ($|I_1 \cap I_2|$).

Currently, there are two types of private set intersection protocols: 1) Specially-designed protocols, which are faster and more efficient. 2) Generic protocols based on secure computation protocol. Despite the fact that generic protocols have more overhead than specially-designed protocols, they could be adjusted to compute any functions [Pinkas *et al.*, 2015]. Also, the result of specially-designed PSI protocols provides more auxiliary information (returning the intersection set) than generic protocols (returning the size of intersection set), Therefore we choose to implement a secure two-party computation protocol to provide more security.

$$\begin{aligned} & f(I_c^{|l_c|}, I_i^{|l_i|}) \\ =& f(\{x_1, ..., x_{l_c}\}_{I_c}, \{x_i, ..., x_{l_i}\}_{I_i}) \\ =& \textstyle\sum_{k=1}^{l_i} \sum_{j=1}^{l_c} \mathrm{EQ}(\{[x_j]\}_{I_c}, \{[x_k]\}_{I_i}) \end{aligned} \tag{1}$$

In the following section, we describe how to use basic boolean gates, namely EQ and ADD gates, to compute the similarity function $f(I_c^{|l_c|}, I_i^{|l_i|})$. Notice that in our generic PSI protocol, EQ and ADD gates are both secure SFE gates.

4.1 Secure Computation

Recall that we want to securely compute the following similarity function, which consists of multiple secure ADD and EQ gates. Theoretically, every function can be constructed by three basic gates: AND, XOR and INV, in this section, we focus on the evaluation of those three basic gates and how to construct EQ and ADD gates using basic gates.

$$\sum_{k=1}^{l_i} \sum_{j=1}^{l_c} \mathrm{EQ}(\{[x_j]\}_{I_c}, \{[x_k]\}_{I_i})$$

Before evaluating the boolean circuit, server S maintains an input $\{[x_k]\}_{I_i} \in \{0,1\}^{32}$ and Client C maintains $\{[x_j]\}_{I_c} \in \{0,1\}^{32}$. For simplicity, we denote a_i as the *ith* bit of private input $\{[x_k]\}_{P_i}$, b_i as the *ith* bit of private input $\{[x_j]\}_{I_c}$, and $\oplus$ as bit operation XOR. Boolean circuit evaluation depends on XOR-based sharing, which means the evaluation of XOR and INV gate are straight-forward and non-interactive. As

Algorithm 1 EQ Gate Construction

Input: $\mathbf{a} \in \{0,1\}^{32}$, $\mathbf{b} \in \{0,1\}^{32}$.
Output: 0 or 1.

Define 32-bit integer **c**, 1 bit integer r.
for every bit $i : 0 \rightarrow 31$ **do**
  $c_i \leftarrow \text{INV}(\text{XOR}(a_i, b_i))$.
  $r \leftarrow \text{AND}(r, c_i)$
end for
return r

Algorithm 2 ADD Gate (32-bit) Construction

Input: $\mathbf{a} \in \{0,1\}^{32}$, $\mathbf{b} \in \{0,1\}^{32}$.
Output: $\mathbf{a} + \mathbf{b}$.

function halfAdder $(\text{in}_1, \text{in}_2, \text{in}_c)$:
  $\text{out}_s = \text{XOR}(\text{in}_1, \text{in}_2)$
  $\text{out}_c = \text{AND}(\text{in}_1, \text{in}_2)$
return $\text{out}_s, \text{out}_c$
end function

function ADD $(\mathbf{a}, \mathbf{b})$:
  Let $c \leftarrow 0$, $\mathbf{r} \leftarrow \{0,1\}^{32}$
  for every bit $i : 0 \rightarrow 31$ **do**
    $r_i, c \leftarrow \text{halfAdder}(a_i, b_i, c)$
  end for
return r
end function

for AND gate we adopted the R-OT$_1^2$ evaluation using pre-computed multiplication triples [Asharov *et al.*, 2013]. Here in this paper we only provide a brief introduction of AND gate evaluation, for those who are interested in this procedure, we strongly recommend the original paper. As for the ADD gate, we implement the work done by [Choi *et al.*, 2011] using those basic gates.

In the following, we describe secret-sharing, reconstruction and the evaluation processes of three basic gates.

- *Sharing Secrets*: Boolean circuit uses XOR-based secret sharing. For each input bit $a_i \in \{0,1\}$, S generates a random bit $\{a_i\}_s = r_i \in \{0,1\}$ and send $\{a_i\}_c = a_i \oplus r_i$ to C, similarly C generates $\{b_i\}_c = r'_i$ and send $\{b_i\}_s = b_i \oplus r'_i$ to S.
- *Reconstruction*: S sends its local compuation result to C, then C compuates $result = \{result\}_s \oplus \{result\}_c$. In HLA matching scenario, server S does not necessarily need to receive the matching result.
- *XOR Evaluation*: S locally computes $\{a_i\}_s \oplus \{b_i\}_s$, C locally computes $\{a_i\}_c \oplus \{b_i\}_c$.
- *INV Evaluation*: S locally computes $\neg\{a_i\}_s$, C does nothing.
- *AND Evaluation*: AND is evaluated using pre-computed boolean multiplication triples $\lambda = \alpha \wedge \beta$ (generated by R-OT$_1^2$). S and C securely compute $e = a \oplus \alpha$ and $f = b \oplus \beta$ using XOR evaluation and reconstruction. And finally S computes $e \cdot f \oplus f \cdot \{a_i\}_s \oplus e \cdot \{b_i\}_s \oplus \{\lambda\}_s$ and C computes $f \cdot \{a_i\}_c \oplus e \cdot \{b_i\}_c \oplus \{\lambda\}_c$

Similar to normal binary gates, those secure gates can be used to construct complex functions. For instance, S and C want to evaluate INV(XOR(a, b)), S and C first share their secrets and then start to evaluate XOR gate, after the evaluation, S would have $\{a_i\}_s \oplus \{b_i\}_s$ and C has $\{a_i\}_c \oplus \{b_i\}_c$. Without reconstructing, they continue to evluate next INV gate, S computes $\neg\{a_i\}_s \oplus \{b_i\}_s$ depending on the previous result and C dose nothing. Finally they reconstruct their secrets and get result $\neg\{a_i\}_s \oplus \{b_i\}_s \oplus \{a_i\}_c \oplus \{b_i\}_c = \neg a \oplus b$. Algorithm 1 and 2 depicts how to build secure EQ and 32-bit ADD gate using basic gates.

4.2 Circuit Performace

As for circuit optimization, we have adopted several GMW optimization methods mentioned in [Schneider and Zohner, 2013] and showed that GMW protocol achieves similar performance as the garbled circuit in low-latency networks. Our test environment is **Ubuntu 16.04.6 LTS** with **Intel(R) Xeon(R) Gold 5118 CPU 2.30GHz** and **64G RAM**. Our tests show EQ gate evaluations based on Yao's garbled circuit and boolean circuit both complete in less than 2ms. Figure 2 shows evaluation time with different number of client's HLA alleles using both boolean and Yao's circuit. It shows boolean circuit is more efficient than Yao's garbled circuit (Even though it is usually seen as best-performing). Furthermore, we compared our solution in the local environment (low-latency) and in real network condition (high-latency) (Figure 3). For this test, our server uses above configuration and our client is **Ubuntu 16.04.6 LTS** with **Intel(R) i7-7700 CPU 3.60GHz** and **16G RAM**.

We also tested our solution under different size of database, result shows that the computation time is proportional to the size of entries in the database. With 5,000 entries in database, the evaluation time is under 1 min.

Figure 2: Time of Evaluation Process for Different Number of Client's HLA Allele Data (Yao vs. Bool)

4.3 Database Shuffling

Until now, we have introduced the secure evaluation of the function $f(I_c^{|l_c|}, I_i^{|l_i|})$, that's to say, server S and client C could securely compute the size of their intersection revealing nothing but their own private input. However, this does not meet our privacy criteria, a semi-honest client C would still be able to attack and retrieve database's entries in polynomial time. Here's our attack: assume there is no restriction

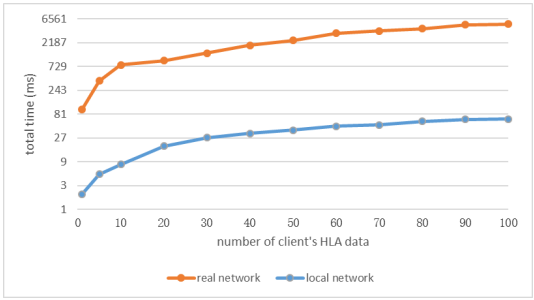

Figure 3: Total Time for Different Number of Client's HLA Allele Data (real network vs. local network)

on client's query data, C may input only one legit HLA allele as P_c. After the computation, C would get a result of either 1 or 0 for each P_i in server's database, this result further helps adversarial C to determine if P_i has this specific HLA allele. By brute-forcing the private input space of HLA alleles, with 2^{32} queries C could successfully retrieve all individual data from database.

$$result_i = f(I_c^{|l_c|}, I_i^{|l_i|}) \tag{2}$$

We notice that the query mechanism is deterministic for client according to Equation 1, the server returns each matching result in the same order they appear in the database, which allows an attacker to associate the result with one specific entry in the database. Therefore, introducing randomness for database could solve this problem, e.g. shuffling the database for every query that server receives. The shuffling mechanism works as follows (Equation 2).

$$result_i = f(I_c^{|l_c|}, I_u^{|l_u|}) \tag{3}$$

where $u = PRNG(secret + query\_ID) + i \pmod{n}$, $PRNG$ is the pseudorandom generator. At the beginning of our protocol, database S uses its own private entropy to generate $secret$. Then whenever S receives a query, it allocates the query with a unique $query\_ID$ and sends it back to C. Next, server S feeds $secret$ and $query\_ID$ into the pseudorandom generator to generate some randomness (from the perspective of C it's random). According to this randomness, S shuffles the database. Therefore at the end of our protocol, C only receives the shuffled matching result and $query\_ID$.

After the HLA matching, if C wants to find the identity of a perfect-matched entry with $query\_ID$ and i, he/she may contact database in person, S would securely reveal the corresponding identity of this entry using $secret$. Normally, the seed generation function can be either a hash function with an entropy or *Password-Based Key Derivation Function 2 (PBKDF2)* function with a password and a salt. We strongly recommend PBKDF2 since it uses iterations to provide more security guarantee than the original hash function, and it has a rather slow calculation process which makes it time-consuming for an attacker to brute-force.

5 Secure Statistics Computation

Researchers have found out that many types of cancer and chronic diseases have genetic predisposition factors [Chakravarti and Little, 2003] [Lander and Schork, 1994]. With the help of large databases, Genome-Wide Association Studies (GWAS) can generalize and discover the rules behind large amounts of genetic data. Most of the GWAS processes combine statistics computation with biological interpretation [Clark *et al.*, 2015] [Denny *et al.*, 2016].

As for other types of genetic data, DNA also has allele polymorphism. A single DNA allele that occurs above a certain degree (normally 5%) within a population will be defined as a variant. Genetic testing can detect variants, however, there are a large number of detected variants we still know little about. GWAS can be implemented to discover potential rules about the relationship between genotype and phenotype. But it requires huge amount of data resource to eliminate random error. The Chi-squared test and odds ratio test are two commonly-used association study methods in GWAS.

In this section, we extend our protocols to cover statistics analysis function and discuss the privacy issues behind the naive implementation. Our work is similar to the work of [Cho *et al.*, 2018], their proposed framework requires raw genotype and phenotype data as input because they work under the assumption of outsourcing computation and clients do not have enough computation power. The lack of local processing increases the workload of their proposed secure computation system. Our work assumes two institutions or hospitals want to run statistics analysis on their private inputs, therefore instead of raw data, we use the following aggregate table as private input (Table 2).

Figure 4: Secure Statistics Computation Scenario

5.1 Odds Ratio and Chi-Squared Test

Odds ratio and Chi-squared Test (χ^2) are both statistics algorithms that quantify the strength of the association between two factors. We provide a privacy-preserving odds ratio and χ^2 test using generic secure function evaluation. The table and equations below introduce the statistics formulas for calculating odds ratios and χ^2.

	case	control
a	a	b
A	c	d

Table 2: OR and Chi's input table

$$\text{Odds Ratio} = \frac{a \times d}{c \times b} \quad \chi^2 = \frac{(a-c)^2}{(a+c)} + \frac{(b-d)^2}{(b+d)}$$

In our designed scenario (Figure 4), there are two parties P_1 and P_2 with reasonable computation power. P_1 holds a private database D_1 and P_2 holds a private database D_2. At the beginning of the protocol, P_1 and P_2 locally compute the aggregate Table 1, denoted as $[D_1]$ and $[D_2]$. We further utilize secure multiplication gate MUL, secure division gate DIV, and *truncate* function from [Catrina and Saxena, 2010] to enable float point secure calculation with fixed-precision.

In this section we do not concentrate on the details of secure MUL, DIV computation since they are not the main contribution of this paper, instead, we present an attack for naive implementation which reveals private inputs and offers a defence method.

5.2 A Possible Attack and Countermeasure

A naive implementation of secure χ^2 and odds ratio computation takes D_1 and D_2 as inputs without any modification, we argue that naive statistics protocol is vulnerable to attacks. Let's say two database P_1 and P_2 use naive method to securely compute the output of OR, P_1 is a curious-but-honest adversary while P_2 remains honest. To carry out the attack, we assume: 1) During our attack, honest party P_2 does not make any changes to its database D_2. 2) Multiple queries are permitted. In the following, we demonstrate our attack on the native secure odds ratio computation.

At the begining of our attack, P_1 holds a private database of $\{a_1, b_1, c_1, d_1\}$, and P_2 holds $\{a_2, b_2, c_2, d_2\}$. P_1 initiates the first round of OR calculation and receives Equation 3 as the result.

$$OR = \frac{(a_1 + a_2) \times (d_1 + d_2)}{(c_1 + c_2) \times (b_1 + b_2)} \tag{4}$$

P_1 then manually adds one self-selected entry (let's say aa) to its database D_1, then in $\{D_1\}^t$, $a_1' = a_1 + 2$. Next P_1 initiates the second round computation and receives the result Equation 4.

$$OR' = \frac{(a_1' + a_2) \times (d_1 + d_2)}{(c_1 + c_2) \times (b_1 + b_2)} \tag{5}$$

By performing the following computation $\frac{OR}{OR - OR'} = \frac{(a_1 + a_2)}{2}$, P_1 will successfully learn the value a_2 and similarly within 4 rounds of execution of the protocol, P_1 will learn all the inputs of P_2. The attack for χ^2 test follows the same routine.

Here's our defence strategy, in each round of calculation, each party (P_1 and P_2) randomly selects 90% of their raw genetic data to participate in local calculation. This means for adequately large database D_1 and D_2, each round of calculation would result in different input table $[D_1]$ and $[D_2]$, violating our attack assumption 1).

6 Discussion

In this paper, we present a practical HLA matching solution using generic secure computation protocol and introduce database shuffling to improve security of our protocols. On the other hand, we present an extensible secure statistics computation protocol which could be extended to other biostatistical algorithms like linkage disequilibrium analysis and linear logistic regression. Further direction could also point to stronger security models, for instance, changing the semi-honest model to covert security with public verifiability. Because the semi-honest model requires strong security assumptions and in genetic analysis, parties could behave maliciously to learn data from other parties without revealing the identity of themselves.

Acknowledgement

We would like to express our thanks to Yuantong Ding who gave valuable advice on proofreading and experiment design. And we also thank Yi Shuai and Yixin Wei who help us with our experiments and biomedical explanation.

A HLA Gene Encoding Table

HLA Gene	ID	HLA Gene	ID	HLA Gene	ID
HLA-A	1	HLA-U	15	HLA-DQA1	29
HLA-B	2	HLA-V	16	HLA-DQB1	30
HLA-C	3	HLA-W	17	HLA-DPB1	31
HLA-E	4	HLA-Y	18	HLA-DPA2	32
HLA-F	5	HLA-DRA	19	HLA-DPB2	33
HLA-G	6	HLA-DRB1	20	HLA-DMA	34
HLA-H	7	HLA-DRB2	21	HLA-DMB	35
HLA-J	8	HLA-DRB3	22	HLA-DOA	36
HLA-K	9	HLA-DRB4	23	HLA-DOB	37
HLA-L	10	HLA-DRB5	24	MICA	38
HLA-N	11	HLA-DRB6	25	MICB	39
HLA-P	12	HLA-DRB7	26	TAP1	40
HLA-S	13	HLA-DRB8	27	TAP2	41
HLA-T	14	HLA-DRB9	28	HFE	42

References

[Asharov *et al.*, 2013] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer and extensions for faster secure computation. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 535–548, 11 2013.

[Beaver *et al.*, 1990] D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 503–513, New York, NY, USA, 1990. ACM.

[Ben-Or *et al.*, 1988] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, pages 1–10, New York, NY, USA, 1988. ACM.

[C. Yao, 1982] Andrew C. Yao. Protocols for secure computation. In *Annual Symposium on Foundations of Computer Science - Proceedings*, pages 160–164, 12 1982.

[Catrina and Saxena, 2010] Octavian Catrina and Amitabh Saxena. Secure computation with fixed-point numbers. In *Financial Cryptography*, 2010.

[Chakravarti and Little, 2003] Aravinda Chakravarti and Peter Little. Nature, nurture and human disease. *Nature*, 421:412–4, 02 2003.

[Cho *et al.*, 2018] Hyunghoon Cho, David J Wu, and Bonnie Berger. Secure genome-wide association analysis using multiparty computation. *Nature Biotechnology*, 36, 05 2018.

[Choi *et al.*, 2011] Seung Choi, Kyung-Wook Hwang, Jonathan Katz, Tal Malkin, and D Rubenstein. Secure multi-party computation of boolean circuits with applications to privacy in on-line marketplaces. *IACR Cryptology ePrint Archive*, 2011:257, 01 2011.

[Clark *et al.*, 2015] Shaunna Clark, Karolina A Aberg, Srilaxmi Nerella, Gaurav Kumar, Joseph McClay, Wenan Chen, Linying Y Xie, Aki Harada, Andrey Shabalin, Guimin Gao, Sarah Bergen, Christina M Hultman, Patrik K.E. Magnusson, Patrick F Sullivan, and Edwin J C G van den Oord. Combined whole methylome and genomewide association study implicates cntn4 in alcohol use. *Alcoholism, clinical and experimental research*, 39, 07 2015.

[Denny *et al.*, 2016] Joshua Denny, Lisa Bastarache, and Dan M. Roden. Phenome-wide association studies as a tool to advance precision medicine. *Annual Review of Genomics and Human Genetics*, 17, 09 2016.

[Erlich and Narayanan, 2013] Yaniv Erlich and Arvind Narayanan. Routes for breaching and protecting genetic privacy. *arXiv e-prints*, page arXiv:1310.3197, Oct 2013.

[Goldreich *et al.*, 1987] Oded Goldreich, S Micali, and Avi Wigderson. How to play any mental game. *STOC '87 Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 218–229, 01 1987.

[Gymrek *et al.*, 2013] Melissa Gymrek, Amy L McGuire, David Golan, Eran Halperin, and Yaniv Erlich. Identifying personal genomes by surname inference. *Science (New York, N.Y.)*, 339:321–324, 01 2013.

[HLA.Alleles.org, 2010] HLA.Alleles.org. Nomenclature for facotrs of the hla system. http://hla.alleles.org/nomenclature/index.html, 2010. Accessed: 2019-05-09.

[Homer *et al.*, 2008] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V Pearson, Dietrich A Stephan, Stanley Nelson, and David Craig. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, 4:e1000167, 09 2008.

[Lander and Schork, 1994] E.S. Lander and Nicholas Schork. Genetic dissection of complex traits. *Science (New York, N.Y.)*, 265:2037–48, 10 1994.

[Lee *et al.*, 2018] Jin-Ku Lee, Zhaoqi Liu, Jason K. Sa, Sang Shin, Jiguang Wang, Mykola Bordyuh, Hee Jin Cho, Oliver Elliott, Tim Chu, Seung Won Choi, Daniel Rosenbloom, In-Hee Lee, Yong Jae Shin, Hyunju Kang, Donggeon Kim, Sun Kim, Moon-Hee Sim, Jusun Kim, Taehyang Lee, and Do-Hyun Nam. Pharmacogenomic landscape of patient-derived tumor cells informs precision oncology therapy. *Nature Genetics*, 50, 09 2018.

[Liu *et al.*, 2018] Siyang Liu, Shujia Huang, Fang Chen, Lijian Zhao, Yuying Yuan, Stephen Starko Francis, Lin Fang, Zilong Li, Long Lin, Rong Liu, Zhang Yong, Huixin Xu, Shengkang Li, Yuwen Zhou, Robert W. Davies, Qiang Liu, Robin G. Walters, Kuang Lin, Jia Ju, and Xun Xu. Genomic analyses from non-invasive prenatal testing reveal genetic associations, patterns of viral infections, and chinese population history. *Cell*, 175:347–359.e14, 10 2018.

[Pinkas *et al.*, 2015] Benny Pinkas, T Schneider, and G Segev. Phasing: Private set intersection using permutation-based hashing. *Proceedings of the 24th Conference on USENIX Security Symposium*, 15:515–530, 01 2015.

[S. Shringarpure and Bustamante, 2015] Suyash S. Shringarpure and Carlos Bustamante. Privacy risks from genomic data-sharing beacons. *American journal of human genetics*, 97, 11 2015.

[Schneider and Zohner, 2013] Thomas Schneider and Michael Zohner. Gmw vs. yao? efficient secure two-party computation with low depth circuits. In *Finacial Cryptography and Data Security*, volume 7859, pages 275–292, 04 2013.

[Shendure *et al.*, 2019] Jay Shendure, Gregory M. Findlay, and Matthew W. Snyder. Genomic medicine–progress, pitfalls, and promise. *Cell*, 177:45–57, 03 2019.

[Vento-Tormo *et al.*, 2018] Roser Vento-Tormo, Mirjana Efremova, Rachel Botting, Margherita Turco, Miquel Vento-Tormo, Kerstin B. Meyer, Jong-Eun Park, Emily Stephenson, Krzysztof Polanski, Angela Goncalves, Lucy Gardner, Staffan Holmqvist, Johan Henriksson, Angela Zou, Andrew Sharkey, Ben Millar, Barbara Innes, Laura Wood, Anna Wilbrey-Clark, and Sarah A. Teichmann. Single-cell reconstruction of the early maternal–fetal interface in humans. *Nature*, 563, 11 2018.