GenMatch: Secure DNA Compatibility Testing

M. Sadegh Riazi*, Neeraj K. R. Dantu*, L. N. Vinay Gattu*, Farinaz Koushanfar†
*Rice University, Houston, TX, USA
†University of California, San Diego, USA
{sadegh, nd17, lg33}@rice.edu, farinaz@ucsd.edu

***Abstract*—We introduce GenMatch, a novel set of techniques based on hardware synthesis, for achieving efficient and scalable privacy-preserving genetic testing. Processing and handling sensitive genome data require methodologies to thwart possible attacks and data theft scenarios. The GenMatch secure genome testing method utilizes Yao's Garbled Circuit (GC) protocol and creates a formulation of the matching problem in a sequential GC format. Our formulation involves private matching of genome data by the GC protocol. Our method reduces the memory footprint of the secure computation such that it can be done in a resource-constrained devices like embedded platforms, rendering the method scalable and time-efficient. Proof-of-concept evaluations are performed on the application of matching Human Leukocyte Antigen (HLA) data for organ and tissue transplant compatibility between recipient and donors. This type of testing also has applications in ancestry testing and genetic matchmaking. HLA data of the recipient is matched with a database of possible donor HLA data while keeping the data from both parties private. Experimental results on real genome data demonstrate the practicability of GenMatch in terms of timing and communication complexity for HLA database in the order of million user profiles.**

***Index Terms*—Privacy-Preserving Computing, DNA Matching, Secure Function Evaluation, Garbled Circuit**

I. Introduction

Whole genome sequencing is a scientific process that is used to determine the complete DNA sequence of an organism [1]. A lot of progress has been made in this area in the past couple of decades. In fact, efforts are being made to commercialize this process because of the potential applications. A number of companies are competing for the market share of cost-effective platform for full genome sequencing [2].

The availability of embedded platforms capable of the full sequencing of a genome with on-board storage of the data in digital hardware, introduces new applications, challenges, and opportunities. On one hand, several new and exciting applications can be realized at a fraction of today's cost. One such emerging application is personalized medicine where the effect of a drug on a person can be tested genetically to determine the drug's compatibility with the person. This can also lead to reducing the risks of side effects and adverse reactions. Another application is genetic compatibility which determines agreeableness of genomes between an organ donor and a recipient. The first generation sequencing companies have mostly failed in this type of testing, mainly due to the high cost associated with these procedures. Furthermore, in the earlier commercial solutions, both partners had to send genome samples to the testing company, which was both inconvenient and privacy invasive.

On the other hand, the privacy requirements for handling sensitive genome data arise serious challenges. Not only the sensitive DNA data reveals important personal information about the individual, but also, the process is irreversible and could cause lifelong irreparable damages to the DNA owner and her relatives sharing similar genes. As a result, it is necessary to devise methods that protect the privacy of individuals interacting with third party companies or research organizations handling genome data. Earlier work in this area explored ways of preserving privacy by anonymizing the data or formulating secure protocols. The current literature falls within one of the two categories: (i) heuristic solutions that lack strong security proofs or guarantees within the standard model [3], [4] or (ii) provably secure solutions that have limited scalability on real DNA datasets [5].

Our work explores practical privacy-preserving DNA testing based on Yao's provably secure Garbled Circuit (GC) protocol. Specifically, we consider Human Leukocyte Antigen (HLA) analysis which is a crucial test in organ transplantation [6]–[9]. First, the raw genome data of the organ receiver (or donor) is processed to obtain required HLA information for performing the privacy-preserving genome test. (In our case, we used the raw genome data from the Human Genome Project [10] database.) Then, the GC protocol is used to compare this HLA data with an existing HLA database of donors (or receivers) to compute the best match for the HLA data while preserving the privacy of all parties involved. We adopt the TinyGarble platform [11] to implement the GC protocol. TinyGarble takes a sequential circuit description of the function that needs to be evaluated securely. We formulate a sequential circuit for performing the HLA data compatibility test and show that it can be efficiently implemented using Verilog. Our explicit contributions are as follows:

- Introduction of the first efficient, practical, and scalable methodology for secure organ and tissue transplantation compatibility test. Our method lever-

978-1-4673-8826-9/16/$31.00 ©2016 IEEE

ages hardware synthesis techniques to formulate the genome matching algorithm for the GC protocol. The low memory footprint of our method allows the first implementation of provably secure matching on embedded devices.

- Design of the first sequential circuit for the purpose of organ compatibility testing. The circuit performs a comparison with one entity in the database at each sequential clock cycle and incurs a circuit size of logarithmic complexity which can scale well for big database sizes.
- Creation of special computational blocks customized for the GC-based DNA matching applications. We also suggest a new method for a cumulative addition where the output bit-length increases with each stage.
- Proof-of-concept implementation of privacy-preserving organ and tissue transplantation compatibility test and demonstrating the scalability of our work by performing a compatibility test on a database in the order of million user profiles.

II. Preliminaries and Definitions

A. Human Genome

A human genome is the complete set of genetic and biological information of a person. It consists of about 3 billion nucleotides of types guanine (G), adenine (A), thymine (T), or cytosine (C). Several active research projects aim at analyzing the genome data to extract specific information or finding a correlation between genome patterns and physical/mental traits.

We focus on a specific part of DNA called the Human Leukocyte Antigen (HLA). HLA genes are the human equivalents of Major Histocompatibility Complex (MHC) genes found in most vertebrates. HLA genes encode proteins that are responsible for regulation of the immune system in humans. In other words, HLA is responsible for determining whether a tissue is native or foreign. HLA is present in chromosome 6 of the genome. There are several classes of this part of the DNA and they have different functions.

The HLA genes that are inherited by an individual on a single chromosome constitute a "haplotype" [12]. Each person receives a pair of HLA genes, one from each of the parents. An example of HLA data of a person is shown in Fig. 1.

HLA mismatch in case of organ transplant is a major cause of transplant rejections and graft-versus-host-disease. Therefore, it is important to match HLA data before transplantation. It should also be noted that HLA data can reveal information about genetic ancestry and predisposition of the person towards certain autoimmune diseases. Apart from this, HLA can also be used for genetic compatibility testing. It was observed that married couples were less likely to share HLA alleles [13].

Paternal inheritance	Maternal inheritance
HLA-A*01:01	HLA-A*02:01
HLA-B*07:02	HLA-B*08:01
HLA-C*07:01	HLA-C*16:01
HLA-DQA*02:01	HLA-DQA*05:01
HLA-DQB*02:01	HLA-DQB*02:01
HLA-DRB*03:01	HLA-DRB*07:01

Fig. 1: Sample HLA template for an individual.

B. Secure Computation

In a two-party privacy-preserving computing, Alice and Bob aim to jointly evaluate a given public function $z = f(x, y)$ where x and y are the private inputs owned by Alice and Bob respectively. At the end of the computation, the output will be available to one party or to both.

In the GC theory, the function $f(.,.)$ must be described as a Boolean circuit where the function arguments (x and y) should be strings of bits. Representing a function as a Boolean circuit may result in a large number of Boolean gates. Each 2-input Boolean gates must be accompanied by calls to a standard cryptographic function (e.g., AES). The only exception is XOR; the authors in [14] have shown that an XOR gate can be garbled with a negligible cost without requiring expensive cryptography. Therefore, the computational cost of GC can be considerably reduced by representing the circuit with the minimum number of non-XOR gates.

Recent advances in this area suggest that the GC protocol can be realized in three stages: (i) circuit garbling, (ii) data exchange, and (iii) circuit evaluation. One of the parties say, Bob, becomes the Garbler, responsible for garbling the circuit. Communication involves both Alice and Bob. Finally, the circuit is evaluated by the non-Garbler party, i.e., Alice here. The computational time and communication are dominated by and also proportional to the total number of garbled tables (non-XOR gates) needed to evaluate the circuit.

III. Related Work

Genetic testing and it's digitization raise some important privacy and ethical concerns. The authors in [15] summarize the issues and put forth the main challenges for the research community. They not only talk about technical issues such as efficiency, usability, and pitfalls of genetic testing, but also about policies that should be made into laws when handling genome data. The work in [16] answers some of the questions in [15] and provides a framework for the secure handling of genome data.

There are different technical approaches to *in silico* privacy-preserving genetic testing. In [17], the authors create android applications for genetic testing methods

such as paternity testing, ancestry testing, and personalized medicine. They are concerned with the efficiency and usability of the applications and conclude that this is an area worth exploring. They use hash function based Private Set Intersection Cardinality (PSI-CA), Authorized Private Set Intersection (APSI) and additively homomorphic encryption of Secure Hamming Distance (SHD) for each of the applications and implement them on an android based platform. They show that comparing the whole genomes for compatibility is both unnecessary and computationally infeasible for a mobile device. This makes the pre-processing an important and aiding step to extract the information that is needed for a specific test. This pre-processing stage helps to carry out the test on a mobile device. They propose P^3MT protocol to test for HLA-B mutation which is important in determining the sensitivity to a drug used in HIV treatment. Our work differs from their work in terms of the framework of the test as well as the application. They work with HLA data but their test is limited to a particular mutation as an example for personalized medicine in contrast with the database-based compatibility test described in this paper.

The work in [5] describes a GC-based approach for measuring the similarity between two genome sequences by performing secure edit distance computation. This computation is performed on raw genome data and is not scalable. It utilizes the full genome which is unnecessary especially when only a small part of the genome data needs to be tested.

The authors of [18] propose homomorphic encryption based identification, paternity and ancestry testing exploiting the Short Tandem Repeat (STR) property exhibited by genetic sequences. Another proposal in [19] uses homomorphic encryption to perform queries on encrypted database of genome sequences while preserving the identities of each of the individuals. This is similar to our setting but the application involves testing for Single Nucleotide Polymorphisms (SNPs) in a database for research purposes involving the whole genome sequences without pre-processing. The authors in [20] describe a similar situation but consider an implementation of a privacy-preserving forensic DNA database. Each DNA in the database is encrypted using a part of its DNA sample and can only be decrypted if the person submitting the query has the appropriate key generated from the suspect's DNA. While this works for a forensic database application, it cannot be extended to other genetic testings except for those involving identity protection.

The work in [21] concentrates on searching a finite length DNA fragment in another DNA template protecting the privacy of both parties involved. They designed a protocol that allows an oblivious evaluation of a finite state machine which has a linear relation between communication complexity on one side and the number of states and length of input data on the other side. Similar to some of the previously mentioned works, they work with raw genome data making it hard to scale and unnecessarily complex to implement.

In contrast to aforementioned works, we design a scalable database-based secure DNA compatibility testing utilizing the GC protocol. We adopt TinyGarble platform [11] to implement our method and as shown in Section V, our approach is highly efficient and scalable.

IV. Our Approach

A. Scenario

We explore the scenario where a patient needs an organ transplantation and is looking for compatible donors. We assume that the patient has undergone full genome sequencing. The HLA data is obtained from pre-processing the genome data. Fully sequenced genomes are very large files and secure computation on the data as a whole is both infeasible and unnecessary. Instead, we extract the information required for compatibility testing through off-line processing. Assuming we have a database of HLA data of donors, we find the best possible match for the patient's HLA data.

B. Pre-processing

The HLA data needed for our test can be extracted from the fully sequenced genome. Samples of fully sequenced genome data are available through the Human Genome Project to researchers for analysis and application specific usage [10]. This data can be processed using HLA-genotyper [22], a python based library used for HLA typing. This is an offline process as it involves no interaction between the two parties and it is also convenient as it only has to be done once and the obtained data can be stored and used in the future.

C. Boolean Circuit

The extracted data from pre-processing contains a pair of haplotypes. One from the mother and the other from the father. There are 6 pairs of haplotype data, HLA-A, HLA-B, HLA-C, HLA-DQA, HLA-DQB, and HLA-DRB. The Algorithm 1 describes the process of comparison of two HLA data [12], [23], [24].

We use Verilog to characterize the Boolean circuit required for the GC protocol. This circuit calculates the percentage of compatibility between two samples of HLA data and is shown in Fig. 2. This is a sequential circuit which takes the patient's HLA data and also one HLA profile in a database as inputs at each clock cycle. The circuit compares the compatibility among pairwise HLA profile as described in Algorithm 1. The final compatibility is the average of all pairwise HLA type compatibilities. Instead of directly finding the average,

Algorithm 1: Algorithm for computing percentage of HLA compatibility between 2 persons.

Inputs: 6 pairs of HLA data from person 1 (HLA1 [index] [pair]) and 6 pairs of HLA data from person 2 (HLA2 [index] [pair]).

Outputs: Percentage of compatibility between two samples.

1: *total compatibility* $= 0$
2: **for** n = 1 to 6 **do**
3: **if** $HLA1[n][1] == HLA2[n][1]$ **then**
4: **if** $HLA1[n][2] == HLA2[n][2]$ **then**
5: *compatibility* $= 1$
6: **else**
7: *compatibility* $= 0.5$
8: **end if**
9: **else if** $HLA1[n][2] == HLA2[n][1]$ **then**
10: **if** $HLA1[n][1] == HLA2[n][2]$ **then**
11: *compatibility* $= 1$
12: **else**
13: *compatibility* $= 0.5$
14: **end if**
15: **else if** $HLA1[n][1] == HLA2[n][2]$ **then**
16: *compatibility* $= 0.5$
17: **else if** $HLA1[n][2] == HLA2[n][2]$ **then**
18: *compatibility* $= 0.5$
19: **else**
20: *compatibility* $= 0$
21: **end if**
22: *total compatibility* $=$ *total compatibility* $+ \frac{1}{6} \times$ *compatibility*
23: **end for**

we propose an alternative approach, described in Section IV-D, which results in a more efficient computation. The final compatibility is then compared with the previously most compatible profile (pre-)stored in the memory (D-FFs). If this new HLA profile is more compatible than the previous one, this new value along with the index of this new profile will be stored in the memory. To keep track of which index we are comparing at a given time, we have embedded a counter in this circuit which is incremented at each clock cycle by one. Since we compare one profile from the database at each clock cycle, we need to evaluate the circuit for N clock cycles, where N is the number of profiles in the database.

D. Circuit Optimizations and Size Model

The optimizations we describe here are different from the state-of-the-art optimizations that are used for executing the GC protocol in Section IV-E. In Section IV-E, we explain that an XOR gate does not need a garbled table and hence the cost of computation of an XOR gate is negligible. Therefore, the goal is to minimize the number of non-XOR gates in the circuit. Here are some techniques we propose:

Translating floating-point operations into integer operations: As discussed in Section IV-B, we need to calculate the compatibility of pairwise HLA data between the patient and a profile in the database. Possible outcomes for each HLA type comparison can be 0, 0.5 or 1, we can encode these outcomes into 3 binary values 00_2, 01_2 and 10_2. Then only 2 bits are sufficient to represent them. For calculating the final average, we need to do an integer addition instead of the floating-point addition and the division can be omitted since it is a division by fixed number for all comparisons among different profiles (division by 6). This will result in a huge reduction in the circuit size.

Designing special building blocks: Since the goal is to reduce the number of non-XOR gates in the circuit, we need to design each block to get the minimum number of non-XOR gates possible. To achieve this goal we have designed special comparison block. This block has only $b-1$ non-XOR gates for comparing the binary input of length b-bit. At each clock cycle, we need to increment the counter by one and this is fixed for all clock cycles and hence, we have optimized this counter to have the least number of non-XOR gates.

Adder optimization: In order to find the summation of 6 pairwise comparisons between different HLA types, we propose a hierarchical structure as shows in Fig. 2 instead of instantiating the default addition block. This structure is more efficient because it gives us a dynamic bit-length. The bit-length of output for each adder increases as we add the results of more comparisons. As an example, in the worst-case-scenario, for the first addition, we only need to add 10_2 and 10_2 which will result in 2-bit adder and use a carryout bit. As we progress in this structure we need higher bit-length to represent the result and this will give us the most efficient way to do this summation. Also, each adder is specially designed to have the least number of non-XOR gates.

Perfect match signal: We have a 1-bit signal in the circuit which sets to high if we find a perfect match between patient's HLA data and one of the profiles in the bank. Therefore, there is no need to continue the protocol and we can terminate it sooner. The perfect match happens when all HLA types are same. Since every value during the execution of the GC protocol is encrypted, both Garbler and Evaluator need to reveal their shared secret to achieve the actual value of this signal. While revealing this bit will inform us as soon as we have found the best match possible, it will not reveal anything about any HLA data from patient side and from database side.

1) Circuit Size Model: Here we analyze the size of the circuit and present a quantitative formula for its size. There are some parts of the circuit that are not dependent on the database size (N) such as HLA comparison blocks

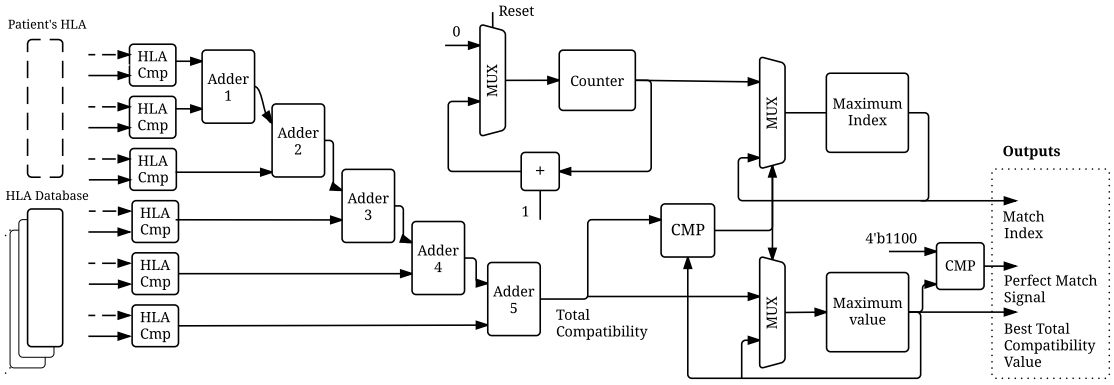

Fig. 2: Architecture of the sequential circuit used in the GC protocol.

and hierarchical structure of adders. However, some parts have a number of non-XOR gates proportional to $log\, N$, including counter and maximum value comparison block. Therefore, the final number of non-XOR gates is a linear function of $log\, N$ plus a constant factor. Fig. 3 proves our analysis and shows the number of non-XOR gates for different number of database size (N). The experimental results deviate slightly from theoretical function due to the heuristic approaches used for synthesizing the circuit. The formula for calculating the number of non-XOR gates is $\alpha \times log\, N + \beta$, where $\alpha = 18.6$ and $\beta = 395$. As we discussed, we need to evaluate this sequential circuit for total number of N clock cycles to compare the patient's HLA data with all of the HLA data in the database. Therefore, total number of garbled tables we need to evaluate, one for each non-XOR gate, is N times the total number of non-XOR gates in the circuit:

$$\text{Total \# of garbled tables} = N \times (\alpha \times log\, N + \beta)$$

Above equation shows that our approach can be easily scaled up to huge database sizes such as million due to the *linearithmic* complexity. Note that in the GC protocol, the computational time and communication are proportional to the total number of garbled tables.

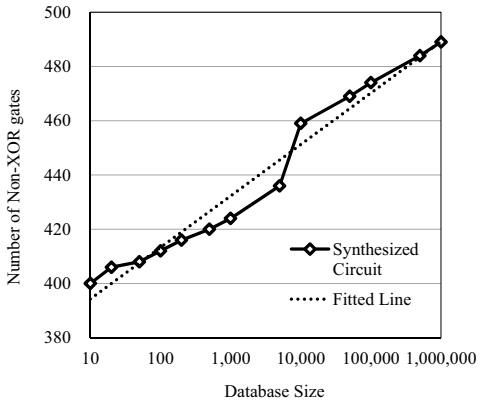

Fig. 3: Number of non-XOR gates of the circuit vs. database size N.

E. Garbled Circuit Protocol Optimizations

We explained the GC protocol in Section II. Here, we list some of the recent optimizations, we have used:

- Free XOR [25]: Kolesnikov and Schneider suggest that XOR gates can be evaluated without generating and sending the garbled table and this will make the XOR gates almost free of cost.
- Half-AND [26]: In this paper, authors introduce a new technique for processing AND gates in the GC protocol. They prove that any AND gate can be garbled and evaluated by a garbled table with only two rows as opposed to the traditional four-row table. Therefore, almost 50% efficiency is achieved for processing AND gates.
- Garbling with fixed-key block cipher [27]: Bellare et al., suggested usage of fixed-key block cipher to implement Hash function used in the GC protocol. Using a processor with AES instruction in its ISA, computing each hash function takes one clock cycle of the CPU.

V. Results

Our experimental results are performed using two processes on Intel Core i7-2600 CPU @ 3.4GHz with 12GB RAM on a 64-bit Ubuntu 14 operating system. The security parameter in our setup (encryption key length) is 128-bit. The circuits are synthesized by the Synopsis Design Compiler. Table I shows the results. Since there is no similar GC-based work with an application performing on a database of genome data, we could not compare our results with previous works.

VI. Conclusion

In this paper, we introduced the first practical and scalable realization of a provably secure HLA matching used in organ transplantation donor compatibility test under the honest-but-curious attack model. Our approach utilizes circuit optimization and logic synthesis for finding a scalable implementation of HLA matching in the

Database Size	# of XORs	# of Non-XORs	Total Gates	Total Garbled Tables	Communication (MBytes)	Time (s)
10	438	400	838	4,000	1.0	0.07
100	447	412	859	41,200	10.5	0.62
1,000	457	424	881	424,000	108.5	5.79
10,000	433	459	892	4,590,000	1,175.0	63.20
100,000	436	474	910	47,400,000	12,134.4	546.09
1,000,000	439	489	928	489,000,000	125,184.0	5,132.25

TABLE I: Number of XOR and non-XOR gates of circuit and total timing and communication for different size of database.

TinyGarble framework. We demonstrated an end-to-end implementation of the system. Our results show that the methodology is highly efficient, and it takes only a few hours to perform the matching on a database of a million participants with pre-processed data. These results are scalable, and way more practical than the state-of-the-art in this field, and they enable a range of new applications for privacy-preserving genetic testing.

ACKNOWLEDGMENT

This research is partially supported by an Office of Naval Research grant (ONR-R17460) and a National Science Foundation grant (CNS-1059416) to Rice University. We thank Professor Emiliano De Cristofaro, Professor Ahmad-Reza Sadeghi, and Ebrahim M. Songhori for their assistance and comments that greatly improved the manuscript. We would also like to show our gratitude to anonymous reviewers for their useful comments.

REFERENCES

[1] P. C. Ng and E. F. Kirkness, "Whole genome sequencing," in *Genetic Variation*. Springer, 2010, pp. 215–226.

[2] Commercialization of full genome sequencing, http://www.genengnews.com/gen-articles/race-to-cut-whole-genome-sequencing-costs/939/.

[3] B. A. Malin, "An evaluation of the current state of genomic data privacy protection technology and a roadmap for the future," *JAMIA*, pp. 28–34, 2005.

[4] Y. Erlich and A. Narayanan, "Routes for breaching and protecting genetic privacy," *Nature Reviews Genetics*, pp. 409–421, 2014.

[5] S. Jha, L. Kruger, and V. Shmatikov, "Towards practical privacy for genomic computation," in *S&P*, May 2008, pp. 216–230.

[6] G. Opelz, T. Wujciak, B. Döhler, S. Scherer, and J. Mytilineos, "Hla compatibility and organ transplant survival. collaborative transplant study." *Reviews in immunogenetics*, pp. 334–342, 1998.

[7] J. M. Cecka, "The unos scientific renal transplant registry–ten years of kidney transplants." *Clinical transplants*, pp. 1–14, 1996.

[8] J. Sierra, B. Storer, J. A. Hansen, J. W. Bjerke, P. J. Martin, E. W. Petersdorf, F. R. Appelbaum, E. Bryant, T. R. Chauncey, G. Sale, J. E. Sanders, R. Storb, K. M. Sullivan, and C. Anasetti, "Transplantation of marrow cells from unrelated donors for treatment of high-risk acute leukemia: The effect of leukemic burden, donor hla-matching, and marrow cell dose," *Blood*, pp. 4226–4235, 1997.

[9] D. Speiser, J. Tiercy, N. Rufer, C. Grundschober, A. Gratwohl, B. Chapuis, C. Helg, C. Loliger, M. Siren, E. Roosnek, and M. Jeannet, "High resolution hla matching associated with decreased mortality after unrelated bone marrow transplantation," *Blood*, pp. 4455–4462, 1996.

[10] . G. P. Consortium *et al.*, "An integrated map of genetic variation from 1,092 human genomes," *Nature*, pp. 56–65, 2012.

[11] E. M. Songhori, S. U. Hussain, A.-R. Sadeghi, T. Schneider, and F. Koushanfar, "Tinygarble: Highly compressed and scalable sequential garbled circuits," in *IEEE S & P*, 2015.

[12] The Science Behind HLA, https://www.seattlecca.org/client/documents/The-Science-Behind-HLA-Typing_8460_0.pdf.

[13] C. Ober, L. R. Weitkamp, N. Cox, H. Dytch, D. Kostyu, and S. Elias, "Hla and mate choice in humans," *The American Journal of Human Genetics*, pp. 497–504, 1997.

[14] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, "Secure two-party computation is practical," in *ASIACRYPT*. Springer, 2009, pp. 250–267.

[15] E. Ayday, E. D. Cristofaro, J. Hubaux, and G. Tsudik, "The chills and thrills of whole genome sequencing," *CoRR*, 2013. [Online]. Available: http://arxiv.org/abs/1306.1264

[16] M. Naveed, E. Ayday, E. W. Clayton, J. Fellay, C. A. Gunter, J. Hubaux, B. A. Malin, and X. Wang, "Privacy and security in the genomic era," *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1405.1891

[17] E. De Cristofaro, S. Faber, P. Gasti, and G. Tsudik, "Genodroid: are privacy-preserving genomic tests ready for prime time?" in *Proceedings of the 2012 ACM workshop on Privacy in the electronic society*. ACM, 2012, pp. 97–108.

[18] F. Bruekers, S. Katzenbeisser, K. Kursawe, and P. Tuyls, "Privacy-preserving matching of dna profiles," Cryptology ePrint Archive, Report 2008/203, 2008, http://eprint.iacr.org/.

[19] M. Kantarcioglu, W. Jiang, Y. Liu, and B. Malin, "A cryptographic approach to securely share and query genomic sequences," *T-ITB*, pp. 606–617, Sept 2008.

[20] P. Bohannon, M. Jakobsson, and S. Srikwan, "Cryptographic approaches to privacy in forensic dna databases," in *Public Key Cryptography*, ser. Lecture Notes in Computer Science, H. Imai and Y. Zheng, Eds. Springer Berlin Heidelberg, 2000, pp. 373–390. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-46588-1_25

[21] J. R. Troncoso-Pastoriza, S. Katzenbeisser, and M. Celik, "Privacy-preserving error resilient dna searching through oblivious automata," in *CCS*. ACM, 2007, pp. 519–528.

[22] J. J. Farrell, "The prediction of hla genotypes from next generation sequencing and genome scan data," Ph.D. dissertation, BOSTON UNIVERSITY, 2014.

[23] HLA Compatibility Calculator, http://www.hiv.lanl.gov/content/immunology/hla/hla_compare.html.

[24] Kidney Transplantation, https://web.stanford.edu/dept/HPS/transplant/html/hla.html.

[25] V. Kolesnikov and T. Schneider, "Improved garbled circuit: Free xor gates and applications," in *Automata, Languages and Programming*. Springer, 2008, pp. 486–498.

[26] S. Zahur, M. Rosulek, and D. Evans, "Two halves make a whole," in *EUROCRYPT*. Springer, 2015, pp. 220–250.

[27] M. Bellare, V. T. Hoang, S. Keelveedhi, and P. Rogaway, "Efficient garbling from a fixed-key blockcipher," in *S&P*. IEEE, 2013, pp. 478–492.