Simpler Protocols for Privacy-Preserving Disease Susceptibility Testing

George Danezis
University College London
g.danezis@ucl.ac.uk

Emiliano De Cristofaro
University College London
e.decristofaro@ucl.ac.uk

ABSTRACT

This short paper presents a preliminary description of two new protocols for privacy-preserving disease susceptibility testing, following the model proposed by Ayday et al. in [4]. We show that an alternative encoding of the patient's SNPs can simplify private computations, and make patient-side computation on a trusted smartcard device extremely efficient. To support larger tests, we propose a second protocol variant based on secret sharing that is also simpler than the original proposal, and relies on more efficient primitives.

1. INTRODUCTION

Over the last two decades, advances in Whole Genome Sequencing (WGS) – the suite of technologies used to determine the complete DNA sequence of an organism – have been quite exceptional, boasting a drop in costs, and an improvement in throughput, significantly faster than what Moore's Law would predict [16, 17]. The first full sequencing of a human genome was completed in 2003, after a 13-year, $3B-worth research collaboration involving more than 20 institutions worldwide—the "Human Genome Project" [18]. Numerous companies and research institutions have since then competed in a race toward more and more affordable and accurate technologies, with prices plummeting to hundreds of thousands of dollars already in 2008. The long-anticipated $1,000 threshold was breached by the San Diego-based company Illumina early 2014.

Inexpensive WGS facilitates the collection of a large number of digitized genomes and it is considered a key enabler of research in genomics. For instance, it facilitates the discovery of correlations between common genetic variants and traits, such as disease predisposition or response to treatment. Also, full sequencing become affordable, genome sequencing (and testing) will soon be available to the masses, enabling the advent of a new era of *personalized medicine* [13], where medical care can be tailored to every patient's genetic makeup. Genomic tests are already routinely used to predict patients' response to several drugs and to help doctors assess the right therapy for millions of HIV, cancer, and leukemia patients. Naturally, the availability of a patient's fully sequenced genome will make it easier to run complex tests, in a matter of seconds, *in silico* (i.e, in computation), as opposed to more expensive and slower *in-vitro* procedures.

Privacy Issues with WGS. Unsurprisingly, however, genomic data sharing and dissemination raise a number of important privacy, ethical, and legal concerns. The human genome not only uniquely and irrevocably identifies its owner, but also contains information about ethnic heritage, predisposition to diseases (e.g., Alzheimer's, breast and ovarian cancer, etc.) and conditions (including mental disorders, such as schizophrenia), and many other phenotypic traits [6, 11, 12]. As the human genome contains detailed information about ethnicity and susceptibility to somatic and mental conditions, its disclosure is often associated to the fear of *eugenism* – i.e., genetic discrimination – which bears potentially dreadful implications on social dynamics as well as hiring and health insurance practices. Due to its hereditary nature, disclosing one's genome essentially implies disclosing the genomes of close relatives, as demonstrated, among others, by a recent paper by Humbert et al. [14]. Masking sensitive portions of the genome, e.g., mutations that indicate disease predisposition, is essentially impossible as correlation (specifically, *linkage disequilibrium*) between one or multiple genetic mutations can often be used to reconstruct the "redacted" features [9].

Privacy-preserving Genome Testing. As a result, the research community has started to propose cryptographic techniques to support privacy-preserving in-silico testing on whole genomes. Baldi et al. [5] are the first to support private paternity, ancestry, and personalized medicine tests on whole genomes, and later develop the GenoDroid framework to deploy privacy-friendly testing apps on Android [8]. Ayday et al. [2, 3] also focus on the privacy of personal use of genomic data (e.g., in medical tests and personalized medicine methods), and propose methods for protecting user's genomic privacy by considering the statistical relationship between the variants. Other recent relevant works include [7, 10, 15, 19].

Roadmap. This short paper builds on the work by Ayday, Raisaro, Hubaux, and Rougemont [4], who present a protocol (which we denote as **ARHR13**) for assessing genetic susceptibility to a given disease in a privacy-preserving way. Susceptibility is determined by computing a weighted average, based on the patient's Single Nucleotide Polymorphisms (SNPs)[1] and some importance factors (aka markers) of each SNP (which possibly constitute the test's "secret sauce" and a pharmaceutical company's trade secret).

Contributions. We revisit ARHR13's proposal for and show that a much simpler, yet equivalent, encoding can be used for the SNPs. Based on this intuition, we propose two protocol variants: the first relies on the user-side smartcard to process (part of) the computation, and the second – on infrastructure servers. While these protocols fulfill all requirements of the original proposal, they are simpler and possibly significantly more efficient.

2. PRIVACY-PRESERVING DISEASE SUSCEPTIBILITY TEST

We now review the ARHR13 protocol [4] and introduce the intuition behind a simpler SNP encoding.

[1]SNPs occur when a single nucleotide (A, C, G, or T) differs between members of the same species or paired chromosomes of an individual and are sometimes associated with disease predisposition and response to treatment.

2.1 The ARHR13 Protocol

In [4], Ayday, Raisaro, Hubaux, and Rougemont aim to privately assess the susceptibility of a patient P to a disease X as a weighted average involving, for each of patient's SNP_i, an importance factor, C_i, and a SNP-dependent weight, $\Pr[X|\text{SNP}_i \in \{0, 1, \text{ or } 2\}]$, where 0, 1, 2 denote, respectively, the presence of the SNP in no, one, or both chromosomes[2]:

$$S = \frac{\sum_i C_i \cdot \Pr[X|\text{SNP}_i]}{\sum_i C_i}$$

Their model, illustrated in Figure 1, assumes the presence of a Certified Institution (CI) which receives genetic samples, sequences them, and produces an encrypted encoding of all possible SNPs. The encrypted sequence is then stored on another (cloud-based) entity called Storage and Processing Unit (SPU). Patients are issued with a smartcard, that contains part of the secret key necessary to decrypt the genetic information. When a test is to be performed by, e.g., a doctor at a medical centre (denoted as MC), the MC initiates a protocol involving the SPU (holding the encrypted data), the subject's smartcard (holding a secret key), and the MC itself (which holds the weights, also encrypted for trade secrecy reasons). The protocol is so that the results of the test are calculated over encrypted genetic data. That is, neither the SPU nor the MC obtains the exact genetic data of the patient, making them resilient to malicious insiders, hackers, and data loss.

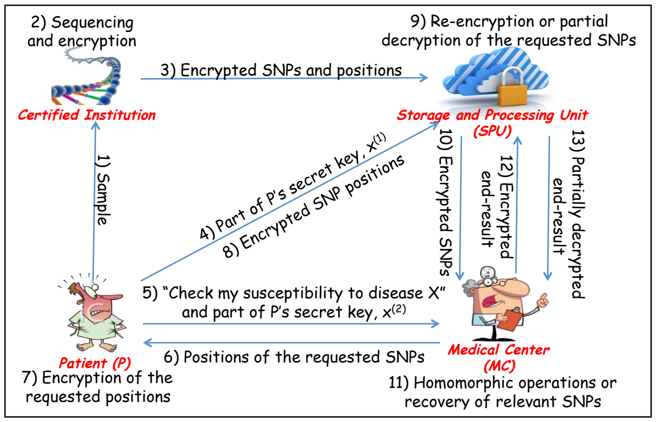
2) Sequencing and encryption
9) Re-encryption or partial decryption of the requested SNPs
3) Encrypted SNPs and positions
Certified Institution
Storage and Processing Unit (SPU)
1) Sample
4) Part of P's secret key, $x^{(1)}$
8) Encrypted SNP positions
10) Encrypted SNPs
12) Encrypted end-result
13) Partially decrypted end-result
5) "Check my susceptibility to disease X" and part of P's secret key, $x^{(2)}$
6) Positions of the requested SNPs
Patient (P)
Medical Center (MC)
7) Encryption of the requested positions
11) Homomorphic operations or recovery of relevant SNPs

Figure 1: ARHR13's privacy-preserving susceptibility testing [4].

• **Step 0:** Cryptographic keys (public and secret keys) of each patient are generated and distributed to the patients. Symmetric keys are also established between all parties.

• **Step 1:** The patient (P) provides a sample to the Certified Institution (CI) for sequencing.

• **Step 2:** The CI performs sequencing and encrypts patient's real SNP using a symmetric key k_{PC} shared with P. It also encrypts patient's real and potential SNP positions under P's public key, using the modified Paillier cryptosystem in [1] (which, besides additive homomorphism, also supports proxy re-encryption). P's public key is denoted as $(n, g, h = g^x)$, where the strong secret key is the factorization of $n = pq$, the weak secret key is $x \in [1, n^2/2]$. The encryption of a real SNP_i, under P's public key, results in $E(\text{SNP}_i, g^x)$ and $E(\text{SNP}_i^2, g^x)$. (The second ciphertext is needed to carry out homomorphic operations later on.) Also, the CI encrypts an arbitrary position v_0 for every potential SNP, using k_{PC}.

• **Step 3:** The CI sends the encrypted SNPs of P to the SPU

• **Step 4:** The patient's weak secret key x is divided into two shares: $x^{(1)}$ and $x^{(2)}$ (such that $x = x^{(1)} + x^{(2)}$). Then, $x^{(1)}$ is given to the SPU and $x^{(2)}$ to the MC in the next step. Note that, thanks to the proxy re-encryption property, an message encrypted under P's public key can be partially decrypted by the SPU using $x^{(1)}$, and then decrypted at the MC using $x^{(2)}$ to recover the original message.

• **Step 5:** The MC wants to conduct a susceptibility test on P to a particular disease X, and P provides the other part of his secret key $x^{(2)}$ to the MC.

• **Step 6:** The MC tells the patient the positions of the SNPs required for the test and obtains his consent to run the test.

• **Step 7:** The patient smartcard encrypts each requested position using the symmetric key shared with the CI.

• **Step 8:** The patient sends the SPU the encrypted positions of the requested SNPs.

• **Step 9:** The SPU receives each requested position in an encrypted form. If the patient has a real SNP at the requested position, the SPU retrieves the encrypted SNP at the corresponding (encrypted) location. Otherwise, the SPU retrieves an encryption of v_0. Then, one of two possible scenarios occur:

(a) If the end-result is to be computed (by the MC), the retrieved SNPs are re-encrypted at the SPU under the P's public key. An encrypted SNP (using a random $r \in [1, n/4]$) is re-encrypted, under the same public key, by using a new random number $r_1 \in [1, n/4]$.

(b) If relevant SNPs are requested (by the MC), the SPU partially decrypts the retrieved SNPs by using $x^{(1)}$ following a proxy re-encryption protocol.

• **Step 10:** Re-encrypted (or partially decrypted) SNPs are sent to the MC (by the SPU) in the same order as they are requested in Step 6.

• **Step 11:** One of two possible scenarios occur:

(a) If the end-result is to be computed (by the MC), the MC computes P's total susceptibility for disease X by using the homomorphic properties of the cryptosystem.

(b) If relevant SNPs are requested (by the MC), the MC decrypts the message received from the SPU by using $x^{(2)}$ and recovers the relevant SNPs.

If the end-result is computed by the MC:

• **Step 12:** The MC sends the encrypted end-result to the SPU.

• **Step 13:** The SPU partially decrypts the end-result using $x^{(1)}$ by following a proxy re-encryption protocol and sends it back to the MC.

• **Step 14:** The MC decrypts the message received from the SPU by using $x^{(2)}$ and recovers the end-result.

2.2 A simpler encoding of SNPs

Our first observation is that each SNP_i may take on 3 distinct values namely 0, 1 or 2. The original ARHR13 scheme encodes those integers directly. On the contrary, we propose to encode each SNP as a 3 bit binary vector, with a value of 1 corresponding to the observed SNP, and the value of zero at other positions (for example SNP_i = 0, 1 or 2 would be represented as 100, 010 or 001, respectively). We then note that, for each binary indicator variable $I_j \in \{0, 1\}$ representing a possible value at a certain $\text{SNP}_i = \alpha$, we can associate a weight $w_j = C_i \cdot \Pr[X|\text{SNP}_i = alpha]$. For a test constant $Z = \sum_i C_i$, the computation reduces to:

$$S = \frac{\sum_j w_j I_j}{Z}$$

[2]If at least one allele carries the mutation, the SNP is denoted as *real*.

The encoding of SNPs as binary indicator variables removes the need for non-linear operations such as squarings that were used in the ARHR13 protocol. The computation is reduced to a simple sum of pairs of secret values – one provided by the pharmaceutical company representing the test (w_j) and one provided by the SPU and representing the genome of the user (I_j). We note that the encodings of the SNP as $\text{SNP}_i \in \{0, 1, 2\}$ or as indicator variables $I_j \in \{0, 1\}$ contain exactly the same information – and given the one in clear, the other may be produced.

Based on this intuition, in the next two sections, we present two protocol variants that leverage the simpler but equivalent encoding of the computation. The first protocol relies on the user-side smartcard to process (part of) the computation, while the second uses infrastructure servers.

3. PRIVATE TESTING PROTOCOL BASED ON SMARTCARDS

We now present our first protocol for privacy-preserving susceptibility testing, aiming to reduce the complexity of the ARHR13 protocol [4]. Specifically, we show how the computation on the private SNPs may be facilitated by some user-held trusted hardware such as a smartcard.

Assumptions. We assume that the patient's SNPs are encrypted, upon sequencing, by the Certified Institution (CI) using standard symmetric encryption (e.g., AES in CBC mode, with a MAC using an encrypt-then-MAC mode), under a key K. The genetic data is also to be signed by the CI to certify its origin. Like in ARHR13, the encrypted SNPs can be stored at the SPU or with the patient's, and may also be provided to the Medical Center (MC) during test execution. Naturally, we recommend the enforcement of some access control mechanism to protect data from unauthorized access even though it is encrypted. Finally, we assume that K is stored on a smartcard provided to the user.

Protocol Sketch. Our goal is to design a protocol that supports the calculation of the disease predisposition, in such a way that genetic data (i.e., the SNPs) is never *in the clear* outside the smartcard, and the trade secrets of the MC (weights associated to the test) are also kept confidential.

To facilitate the computation, the MC encrypts the weights using an Elliptic Curve based El-Gamal (ECC) cryptosystem. Specifically, each weight w_j is encrypted as a tuple of elements $E_x[w_j] = W_j = (k_j \cdot G, (x \cdot k_j) \cdot G + w_j \cdot H)$ where G and H are public generators on the elliptic curve, k_j are fresh secrets and x is the private key only known to the MC. This El-Gamal variant exhibits a simple additive homomorphism, in that pairwise point addition of cipher texts yields an encryption of their sum ($E_x[a]+E_x[b] = E_x[a+b]$).

Whenever a computation is to be performed, the MC provides the subject's smartcard with the encrypted weights W_j as well as an encryption of zero ($E[0]$). The smartcard initializes a register R with the a re-randomized encryption of zero, i.e., $R = E[0]^k$, for a random k. Then, for each value of W_j and I_j read from its environment, it updates the register R as:

$$R' = R + I_j \cdot W_j$$

Since the value of I_j is binary, each step of the computation either involves a single elliptic curve point addition or none. Furthermore, only the SNP positions for which W_j weights have been provided have to be considered for accumulation. In case only a subset of all possible W_j are provided by the pharmaceutical company, others positions do not need to be processed.

When all weights W_j have been processed their signature is checked to ensure they represent a valid test. The accumulated value R is then output from the smartcard, and sent to the MC for decryption. To decrypt a tuple $R = (A, B)$, on input the private key x, the MC computes $H^S = B - x \cdot A$. Since the discrete logarithm problem is hard in the elliptic curve group, recovering the value of S requires using precomputed tables of some maximal size of S.

Efficiency. As smartcards are resource-constraint devices, some care needs to be taken to ensure the protocol runs in reasonable time. Note that the protocol does not require any expensive elliptic curve point multiplications, since it leverages the binary nature of I_j to only perform (cheaper) additions. Modern smartcards can perform thousands of elliptic curve additions a second. Therefore, we expect that the actual bottleneck comes from the I/O of loading the weights into the smartcard, as well as loading and decrypting the I_j values. The internal state of the smartcard comprises a symmetric key K, a handful of elements, and a couple of accumulated hashes – all operations are performed in a streaming manner as data is received.

Note that the MC only needs to store a key, and perform a single decryption per test. The tables used for decryption can be re-used even if the private key is rotated.

Finally, observe that the potentially large number of encrypted SNPs, as well as the encrypted weights W_j, can be pre-fetched or downloaded ahead of time, and can be stored on untrusted storage devices.

4. SECRET-SHARING BASED PROTOCOL

We now describe our second protocol, which, based on secret-sharing, further improves on the original ARHR13 solution.

Despite the few point additions, our first, smartcard-based, protocol may be too inefficient for large tests due to I/O and/or bandwidth constraints of trusted devices or user equipment. Thus, we propose a protocol that makes use of two distinct parties, assumed not to collude with each other. These two parties can be embodied by the SPU and the MC, which in the setting of the original protocol are also assumed not to be colluding and entrusted with a similar burden of computation.

Protocol Sketch. We assume that the CI produces and provides the patient with a smartcard containing the private key y corresponding to a public key $y \cdot G$, where G is a public point on a secure elliptic curve. The public key is then used to output an El-Gamal encrypted stream of all indicator variables I_j corresponding to the patient's SNPs. Each ciphertext is a pair of elements:

$$U_j = (k_j \cdot G, k_j \cdot y \cdot G + I_j \cdot H),$$

where H is a random public point on the elliptic curve.

The weights w_j are also encoded by the pharmaceutical company as a sequence of random values u_j and v_j under the constraint that $u_j + v_j \equiv w_j \mod q$ where q is the order of the group formed by the elliptic curve. Knowledge of either sequences leaks no information about the secret weights w_j. The sequences u_j and v_j are distributed to SPU and MC respectively, who are trusted to not collude to uncover w_j.

The test computation proceeds as follows:

1. The encrypted SNPs, U_j, are download by the SPU and the MC.

2. The SPU and the MC use the shares of the secret weights to compute the following sums over elliptic curve points:

$$A = \sum_j u_j \cdot U_j \qquad B = \sum_j v_j \cdot U_j,$$

where addition denotes pairwise addition of two elliptic curve points, i.e., the elements of the ciphertext.

3. The ciphertext A and B are sent to the patient's smartcard and used to compute $D = A + B$, which is decrypted using the secret y inside the smartcard. This yields an element $H^S = H^{\sum_j w_j \cdot I_j}$, and the test result S can be recovered through the use of a lookup table, either at the patient or using a service at the MC.

Observe that this protocol, compared to the first one presented in Section 3, offers some advantages: since the secret in the smartcard is only used at the end of the protocol, disease susceptibility tests may be pre-computed once the tests are first designed, and then decrypted when the smartcard is provided by the user as a way of authorizing the disclosure of the result. Until that point, no genetic information is leaked. Also note that the scheme may be generalized to any number of trusted parties and shares, or collapsed into a simpler computation if, e.g., a pharmaceutical company acts as a trusted service to facilitate the computation.

Efficiency. Both the SPU and the MC have to perform a number of full elliptic curve multiplications to compute the cipher texts A and B. However, the smart card (which is assumed to be resource-constraint) only needs to perform a few additions and a single El-Gamal decryption to decrypt the result, making this protocol more efficient both in terms of computation and I/O.

5. CONCLUSIONS

This short paper presented a preliminary description of two protocols for privacy-preserving disease susceptibility testing, following the model proposed in ARHR13 [4]. We introduced an alternative but equivalent representation of SNP information, which allows the ARHR protocol to be reduced to the problem of summing products of secrets. This can either be done by relying on a smartcard at the patient's side, or using secret sharing on infrastructure servers. Both protocols ensure that genetic information is always encrypted when in transit or on general purpose hardware, and use a secret within a smart card to unlock the result of the test. Since the numerical domain of the test results is small, we used a simpler and more efficient El-Gamal scheme on elliptic curves, instead of the more expensive Paillier cryptosystem (which requiring computation on $N^2 = (p \cdot q)^2$ which are slow and produce large ciphertexts). In future work, we will study the exact costs of both protocols and their performance on appropriate embedded and server platforms.

References

[1] G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *TISSEC*, 9(1), 2006.

[2] E. Ayday, J. Raisaro, P. McLaren, J. Fellay, and J. Hubaux. Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data. In *HealthTech*, 2013.

[3] E. Ayday, J. L. Raisaro, U. Hengartner, A. Molyneaux, and J.-P. Hubaux. Privacy-preserving processing of raw genomic data. In *DPM*, 2013.

[4] E. Ayday, J. L. Raisaro, J.-P. Hubaux, and J. Rougemont. Protecting and Evaluating Genomic Privacy in Medical Tests and Personalized Medicine. In *WPES*, 2013.

[5] P. Baldi, R. Baronio, E. De Cristofaro, P. Gasti, and G. Tsudik. Countering GATTACA: Efficient and Secure Testing of Fully-Sequenced Human Genomes. In *CCS*, 2011.

[6] T. Canli. The emergence of genomic psychology. *Nature*, 8, 2007.

[7] Y. Chen, B. Peng, X. Wang, and H. Tang. Large-Scale Privacy-Preserving Mapping of Human Genomic Sequences on Hybrid Clouds. In *NDSS*, 2012.

[8] E. De Cristofaro, S. Faber, P. Gasti, and G. Tsudik. GenoDroid: Are Privacy-Preserving Genomic Tests Ready for Prime Time? In *WPES*, 2012.

[9] Y. Erlich and A. Narayanan. Routes for breaching and protecting genetic privacy. *Nature Reviews Genetics*, 15(6):409–421, 2014.

[10] S. E. Fienberg, A. Slavkovic, and C. Uhler. Privacy preserving GWAS data sharing. In *ICDMW*, 2011.

[11] J. Fowler, J. Settle, and N. Christakis. Correlated genotypes in friendship networks. *Proceedings of the National Academy of Sciences*, 108(5), 2011.

[12] M. Fumagalli et al. Parasites represent a major selective force for interleukin genes and shape the genetic predisposition to autoimmune conditions. *Experimental Medicine*, 206(6), 2009.

[13] G. Ginsburg and H. Willard. Genomic and Personalized Medicine: Foundations and Applications. *Translational Research*, 154(6), 2009.

[14] M. Humbert, E. Ayday, J.-P. Hubaux, and A. Telenti. Addressing the Concerns of the Lacks Family: Quantification of Kin Genomic Privacy. In *CCS*, 2013.

[15] A. Johnson and V. Shmatikov. Privacy-preserving data exploration in genome-wide association studies. In *KDD*, 2013.

[16] NHGRI. DNA Sequencing Costs – Data from the NHGRI Large-Scale Genome Sequencing Program. http://www.genome.gov/sequencingcosts, 2012.

[17] E. Singer. Democratizing DNA Sequencing. http://www.technologyreview.com/biomedicine/26850, 2012.

[18] The Human Genome Project. http://web.ornl.gov/sci/techresources/Human_Genome/index.shtml.

[19] R. Wang, X. Wang, Z. Li, H. Tang, M. Reiter, and Z. Dong. Privacy-preserving genomic computation through program specialization. In *CCS*, 2009.