Privacy-Preserving Genome-Aware Remote Health Monitoring

Yanmin Gong*, Chi Zhang†, Yaodan Hu‡ and Yuguang Fang‡
*School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078, USA
†School of Information Science and Technology, University of Science and Technology of China, Heifei 230026, China
‡Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
Email: yanmin.gong@okstate.edu, chizhang@ustc.edu.cn, cindy.hu@ufl.edu, fang@ece.ufl.edu

***Abstract*—Using genetic profiles of individuals for tailored diagnosis and treatment has great promise in the healthcare industry. Despite of the rapid growth in genome-aware medicine, genome-aware health monitoring has not been studied as well. A major stumbling block is the privacy issues of such applications. In addition to privacy concerns in a traditional health monitoring system, i.e., the privacy of users' biomedical sensing data and the protection of the proprietary health monitoring program, severe privacy concerns arise when users' genomic data are integrated into the health monitoring program due to the re-identification and phenotype attacks based on the DNA profile and the relevance of DNA information in a family. In this paper, we investigate these privacy risks and propose a privacy-preserving approach for genome-aware health monitoring. In our approach, users can only learn the diagnostic results based on their genomic and biomedical sensing data, while the the healthcare service provider learns nothing. Security analysis and performance evaluations are conducted to illustrate the effectiveness and efficiency of the proposed approach.**

I. Introduction

Advances in mobile communication and sensing technologies have fostered the rapid growth of mobile health applications in the past years. Among various mobile health applications, remote health monitoring is an important category with a global market valued at $31.4 billion [1]. In a typical mobile health monitoring setting, body sensors collect both biological and physical data of an individual and send the real-time sensing data to a remote healthcare service provider through wireless communication networks. The data can then be used for early diagnosis of physiological or behavioral aberrations, chronic disease tracking, and individual behavior changing, enabling timely intervention and better management of individual health at a reduced cost.

Recently, a new source of personal data, genomic data, has received increasing attention due to the reduced cost of DNA sequencing and is considered as the "next big thing" for disease diagnosis and treatment [2]. Some research has been studied to reveal the relationship between DNA profiles and the treatment of, or predisposition to, diseases such as diabetes, Alzheimer's and certain types of cancer. In fact, commercial companies such as 23andMe [3] and Counsyl [4] have already offered reports on raw genomic data for disease risk or disease treatment guidance. Thus, it is natural to incorporate genomic data in the process of health monitoring. Unlike traditional clinic data, information contained in human genome can no longer be treated as an input to the traditional health monitoring program due to its dimensionality and variances between people. This motivates us to design a remote health monitoring system that is accustomed to individual genomic information rather than a general "one-fits-all" system.

A major challenge in designing a genome-aware remote health monitoring system is the privacy risk. From the perspective of users, they may not want to expose their health data to the healthcare service provider because it may contain sensitive information. While biomedical sensing data reflect current health statuses, activities, and lifestyles of an individual, genomic data reflect a user's phenotype, including predisposition to physical or mental health conditions (such as Alzheimer's disease, cancer, or schizophrenia), and is a irrevocable permanent identifier that carries information of one's family members. Misuse of genomic information may lead to genetic discrimination and other consequences that haven't been fully explored yet. From the perspective of the healthcare service provider, the health monitoring program is proprietary intellectual property and revealing its internal structures or implementation may expose unpatched security vulnerabilities to attackers. While users and the healthcare service provider share the same goal of protecting privacy, they do not trust each other and hence are unwilling to disclose their own data to others.

Privacy-preserving remote health monitoring systems have been studied in [5]–[9], however, none of these works consider using genomic data for diagnosis. Studies on genomic privacy mainly consider secure two-party computation. For example, Ayday et al. develop an architecture for disease susceptibility test and personalized medicine by using homomorphic encryption and proxy re-encryption [10]. In [11], Karvelas et. al. propose a flexible scheme for private processing of genomic sequences by dividing the genomic data in small blocks, which hides both the privacy of the genomic data and the access pattern of the database user. Unlike previous approaches, we address genomic privacy in the remote health monitoring setting, which involves information exchange between multiple entities. Specifically, we present a secure and privacy-preserving framework for genome-aware remote

This work was partially supported by National Science Foundation under grant CNS-1423165. Chi Zhang's work was supported in part by the Natural Science Foundation of China (NSFC) under Grants 61202140 and 61328208, by the Program for New Century Excellent Talents in University under Grant NCET-13-0548, and by the Fundamental Research Funds for the Central Universities under Grand WK2101020006.

health monitoring. In the framework, a branching program is tailored to a user based on his/her genomic information while limiting the sharing of genomic information. With this branching program, the healthcare service provider takes real-time biomedical sensing data of a user as input and returns diagnostic results. We protect the privacy of both the user and the healthcare service provider by ensuring that the user learns nothing about the branching program except his diagnostic results, while the healthcare service provider gains no additional information about the user's health data.

The remainder of this paper is organized as follows. Section II presents the cryptographic primitives used in our scheme. Then, the system model is introduced in Section III. We describe and analyze privacy-preserving protocols for protecting biomedical sensing and genomic data in Section IV. Next, we discuss the security and privacy properties and the performance of our construction in Section V and Section VI, respectively. Finally, we conclude the paper in Section VII.

II. Cryptographical Tools

In this section, we briefly introduce the cryptographic tools used in this paper.

A. Homomorphic Encryption

Homomorphic encryption is a cryptosystem that enables algebraic operations on plaintexts given only their respective ciphertexts. We use a partially homomorphic encryption scheme which allows the ciphertexts of the addition of two plaintexts $E[x_1 + x_2]$ to be computed from the ciphertexts of the two plaintexts $E(x_1)$, $E(x_2)$. There are several partially homomorphic encryption schemes, such as the Paillier cryptosystem [12], Additively Homomorphic ElGamal variant (AH-ElGamal) [13], and Elliptic Curve-based ElGamal cryptosystem (EC-ElGamal) [14].

B. Secure Integer Comparison with Garbled Circuits

Yao is the first to introduce garbled circuit as the solution to secure two-party computation in [15]. In this paper, we use Yao's garbled circuits to implement a sub-protocol for comparing integers in our construction. Consider two parties, Alice and Bob, who wish to securely compute a comparison function. Alice has two L-bit integers β, γ, while Bob has an L-bit integer x. A secure garbled circuit C is generated by Alice to securely compare $x - \beta$ and γ without revealing β and γ to Bob or x to Alice. Specifically, Alice creates a garbled circuit C with L input wires and one output wire. For each input wire l of the circuit, Alice generates two random cryptographic keys ν_l^0 and ν_l^1, encoding 0 and 1, respectively. For the output wire, Alice generates wire keys ρ^0 and ρ^1. After the circuit generation, Alice sends the circuit to Bob for circuit evaluation. To evaluate the circuit, Bob engages Alice in an OT_2^1 protocol where Bob obtains the input wire keys for the l-th wire based on the l-th bit of his input x without revealing x to Alice. Bob then uses these input wire keys to decrypt the circuit gate by gate and obtains either ρ^0 or ρ^1 as the result. This sub-protocol is denoted as COMPARE$(x, \beta, \gamma, \rho^0, \rho^1)$, which returns ρ^0 if $x-\beta \mod 2^L > \gamma$ and ρ^1 if $x-\beta \mod 2^L \leq \gamma$. The above sub-protocol is also known as *privacy-preserving offset integer comparison* [16] or *conditional oblivious transfer with a great than predicate* [17]. Under the assumption that adversaries are computationally bounded, Yao's circuit is currently known as the most efficient method [17].

TABLE I
Notations used throughout this paper

$T = \langle\{p_1, p_2, \ldots, p_I\}, \mathbf{L}, \mathbf{R}\rangle$	Branching program
$\langle a_i, W_i\rangle, 1 \leq i \leq I_d$	Decision node $p_i, 1 \leq i \leq I_d$
w_0, w_1	Optional values for threshold W_i
$\langle d_i\rangle$	Classification node $p_i, i > I_d$
$\mathbf{t} = \{t_1, t_2, \ldots, t_N\}$	Genomic information of the user
$\mathbf{s} = \{s_1, s_2, \ldots, s_M\}$	Reference genomic pattern
$\mathbf{x} = \{x_1, x_2, \ldots, x_K\}$	User's attributes
L	Number of bits of x_k and W_i
pk_u, sk_u	User's public/private key pair
$[\cdot]$	Ciphertext encrypted with pk_u
q	Order of the plaintext group

III. System Model

In this section, we introduce the system model of the proposed genome-aware remote health monitoring system. The list of major notations used in this paper is described in Table I.

A. System Architecture

The core of a remote health monitoring system is a branching program that outputs diagnostic results based on the attribute vectors of users extracted from their biomedical sensing data [16]. The branching program is described as follows. Let $\mathbf{x} = \{x_1, \ldots, x_K\}$ represent a vector of user attributes that will be evaluated by the branching program. A binary branching program T with I nodes can be represented by a triple $\langle\{p_1, p_2, \ldots, p_I\}, \mathbf{L}, \mathbf{R}\rangle$. The first element $\{p_1, p_2, \ldots, p_I\}$ denotes a set of nodes in the branching program. Let I_d be the number of internal nodes in the branching program. When $i \leq I_d$, p_i is an internal node, or "decision node", defined as $p_i = \langle a_i, W_i\rangle$, where $a_i \in [1, I]$ is the index of the attribute in $\mathbf{x}$ and W_i is the threshold with which x_{a_i} is compared at this node. When $x_{a_i} \leq W_i$, the index of the next node is $\mathbf{L}(i)$; otherwise, the index of the next node is $\mathbf{R}(i)$. When $i > I_d$, p_i is a leaf node of the branching program, or "classification node", which is associated with a diagnostic label d_i. To evaluate the branching program over a feature vector $\mathbf{x}$, we start from node 1 and compare the attribute value x_{a_1} with the threshold W_1. Then we go to node $\mathbf{L}(1)$ if $x_{a_1} \leq W_1$ and node $\mathbf{R}(1)$ if $x_{a_1} > W_1$. Repeat the process until we reach a classification node and the evaluation result is the diagnostic label associated with this classification node.

We personalize a branching program by changing the thresholds W_i associated with each decision node. In other words, for different genomic information $\mathbf{t}$, the evaluation order of attributes $\mathbf{x}$ in the branching program is the same, but the comparison threshold for each attribute W_i will be different based on $\mathbf{t}$. In this paper we assume that W_i has two possible values $\{w_0, w_1\}$. Note that in practice W_i may

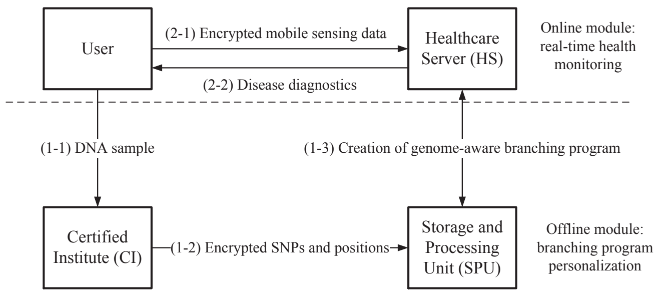
User
(2-1) Encrypted mobile sensing data
(2-2) Disease diagnostics
Healthcare Server (HS)
Online module: real-time health monitoring
(1-1) DNA sample
(1-3) Creation of genome-aware branching program
Certified Institute (CI)
(1-2) Encrypted SNPs and positions
Storage and Processing Unit (SPU)
Offline module: branching program personalization

Fig. 1. System architecture for genome-aware mobile health monitoring.

have more than two possible values depending on different distance metrics such as the edit distance [18], which could be easily implemented as a natural extension of our approach. The genomic information used in this paper is single nucleotide polymorphism (SNP) information which is closely related to disease risks and treatment reactions and is highly useful in personalized diagnostics and treatment [19].

In the proposed genome-aware health monitoring system, there are four entities: the user, healthcare server (HS), certified institution (CI) and storage and processing unit (SPU). The user is the subject who uses health monitoring services, who is continuously monitored by several body sensors. Raw sensing data are transformed into an attribute vector. The HS is run by a health service provider including doctors, insurance companies, diet advisers, athletic coaches or home-care providers, who provides diagnostic decisions based on the attribute vector of the user. The HS owns a genome-aware branching program tailored to the genomic information of the user. The genome is first sequenced and encrypted by a trusted entity CI and then stored at the SPU. The SPU stores the encrypted genomic information for the user and provide his/her genomic data to the HS upon request. The genome-aware branching program is computed during the initialization stage of health monitoring collaboratively by the SPU and the HS. The system architecture is illustrated in Fig. 1. The system consists of an offline module (i.e., *program personalization*) and an online module (i.e., *real-time health monitoring*). In the offline module, the user provides his/her DNA sample to the CI for sequencing (Step 1-1), who sequences the sample and encrypts the genomic data. Then, the encrypted genomic data is sent to the SPU (Step 1-2). After obtaining the consent of the user, the HS collaborates with the SPU to generate a genome-aware health monitoring program based on the user's encrypted genomic data (Step 1-3). In the online module, the user sends an attribute vector extracted from his/her biomedical sensing data to the HS (Step 2-1) and receives diagnostic results from the HS (Step 2-2).

B. Threat Model

We consider the following types of potential adversaries: (i) a malicious attacker at the HS such as a careless/disgruntled employee of the healthcare service provider, who is curious about users' genomic information and would try to infer such information by sending unnecessary queries to the SPU; (ii) a malicious attacker at the SPU, such as a careless/disgruntled employee of the SPU, who is curious about users' genomic information and information about the branching program owned by the HS; and (iii) a curious user who might want to learn the branching program. To defend against aforementioned adversaries, we ensure that neither the HS nor the SPU could learn additional information of a user's genomic information by participating in the *program personalization* module, the SPU learns nothing about the branching program, and the user learns nothing about the branching program except the diagnostic results based on the user's attribute vector.

The adversaries we consider in this paper are assumed to be computationally bounded. Besides, there might be outside adversaries who eavesdrop the communication messages among the user, SPU, CI, and HS. Such adversaries can be defended by using standard symmetric encryption schemes such as DES to secure the communication channels. For ease of presentation we will not discuss these attackers in details.

IV. Privacy-Preserving Genome-Aware Health Monitoring

In this section we first describe a protocol for the offline module to construct a genome-aware branching program which provides privacy protection to both the genomic data and the branching program. Then we illustrates how this genome-aware branching program can be used to evaluate the real-time attribute vector of the user in the online module.

A. Offline Module: Branching Program Personalization

In this subsection, we explain the procedures of the offline module step by step according to Fig. 1. In system setup phase, a public/private key pair of a partially homomorphic cryptosystem $\langle pk_u, sk_u \rangle$ is generated for the user. The public key pk_u will be distributed to all the parties in the system (the CI will use it to encrypt the genomic information of the user $\mathbf{t}$, and the HS will use it to encrypt the reference genomic pattern $\mathbf{s}$), and the private key sk_u will only be distributed to the user.

Step 1-1: The user provides his/her biological sample to the CI for DNA sequencing. The CI sequences the sample and extracts the genomic information of the user. We use the information of SNPs, denoted as $\mathbf{t} = \{t_1, t_2, \ldots, t_N\}, t_n \in \{0, 1, 2\}$, to personalize the branching program. We assume that the positions of SNPs in a human genome are predefined and agreed by all parties in our system so that each element t_n represents the state of a specific SNP.

Step 1-2: CI uses the public key pk_u of the user to encrypt the genomic information $\mathbf{t}$, and send the encrypted genomic information $[t_1], [t_2], \ldots, [t_n]$ to the SPU for storage. Here $[\cdot]$ denotes a probabilistic encryption of a message in the underlying partially homomorphic cryptosystem with user's public key pk_u.

Step 1-3: The HS needs to generate a genome-aware branching program without learning the genomic information of the user. For each decision node i of the branching program, there are M_i SNPs which are used to personalize the threshold W_i, represented by a reference genomic pattern

$\mathbf{s} = \{s_1, s_2, \ldots, s_M\}$. Here s_m represents the state of the m-th SNP in the pattern, and the position of this SNP is represented by pos_m. For each decision node i, the threshold W_i has two possible values w_0 and w_1. If $t_{\text{pos}_m} == s_m, 1 \leq m \leq M$, set W_i to w_0; otherwise,W_i is set to w_1. We omit the subscript i when the context is clear.

Since the genomic information $\mathbf{t}$ of the user is stored in encrypted form at the SPU, the HS and the SPU collaborate to generate the genome-aware branching program. The first step for personalization is to determine whether the genomic information $\mathbf{t}$ matches the reference pattern $\mathbf{s}$ using Algorithm 1.

Algorithm 1 Private-Preserving Personalization of Parameters in the Branching Program

```
Input: Encrypted genomic information $[t_1], [t_2], \ldots, [t_N]\}$,
    reference pattern $\mathbf{s} = \{s_1, s_2, \ldots, s_M\}$, $w_0$, $w_1$
Output: $[W_i]$ {If $[d']$ is an encryption of 0, $W_i = w_0$,
    otherwise $W_i = w_1$}
1: $e_d = [0]$;
2: for $m = 1$ to $M$ do
3:     $r_m \in_R \mathbb{Z}_q$;
4:     $e_d = e_d \cdot ([t_{pos_m}] \cdot [-s_m])^{r_m}$;
5: end for
6: Engage the user in a sub-protocol to transform the encryption of $d$ into the encryption of $d'$, where $d' = 0$ if $d == 0$ and $d' = 1$ if $d \neq 0$
7: Compute $[W_i] = [d'] \cdot w_1 + [1 - d'] \cdot w_0$
8: return $[W_i]$
```

After the HS obtains all the thresholds in encrypted form $[W_i], i = 1 : I_d$, it creates a branching program T' based on the genome-aware branching program T with encrypted thresholds. T' is a randomized transformation and a secure equivalent of the branching program $T = \langle\{p_1, p_2, \ldots, p_I\}, \mathbf{L}, \mathbf{R}\rangle$, $p_i = \langle a_i, [W_i]\rangle$. The conversion from T to T' does not require any interaction with the user, and the HS may generate a large number of T' for the real-time evaluation purpose. Note that in this algorithm, the garbled circuit is designed to handle $L+1$-bit integers in order to handle the overflow caused by negative comparison results.

B. Online Module: Real-Time Health Monitoring

After creating the genome-aware branching program, the HS and the user engage in an online protocol to evaluate the real-time attribute vector of the user, which corresponds to **Step 2-1** and **Step 2-2** in Fig. 1. The user first receives the secure branching program T' from the HS, which is a set of encrypted nodes containing either a garbled circuit or a classification label. The evaluation of T' is similar to the evaluation of a traditional branching program, but the nodes in T' are encrypted and thus need to be decrypted first. Given the first secret key κ_1 for node 1, the user evaluates the garbled circuit C_1 and the output of the circuit contains the index of the next node and the corresponding secret key. In order to prevent the user from getting additional knowledge of the branching program other than the classification result, the attributes being evaluated a_i, the value of the attribute x_{a_i}, and the associated threshold W_i should be kept secret from the user. The nodes in T' have been randomly permuted to hide a_i. Hiding x_{a_i} and W_i, however, is nontrivial because neither the user nor the HS should learn the real values of W_i or x_{a_i}. In our protocol, we introduce a set of $L + L'$ blinding values b_i and c_i to hide x_{a_i} and W_i, respectively. When L' is sufficiently large, these blinding values can hide x_{a_i} and W_i statistically. The protocol is shown in Fig. 2. A major part of the protocol is the oblivious evaluation of garbled circuit C_i, which enables the user to compare x_{a_i} and W_i without learning or revealing their values.

Algorithm 2 Creation of Genome-Aware Branching Program

```
Input: Branching program $T = \langle\{p_1, p_2, \ldots, p_I\}, \mathbf{L}, \mathbf{R}\rangle$ with
    encrypted thresholds $[W_i]$. For $i \leq I_d$, $p_i$ is a decision
    node where $p_i = \langle a_i, [W_i]\rangle$. For $i > I_d$, $p_i$ is a
    classification node containing label $\langle d_i\rangle$.
Output: • Secure genome-aware branching program $T'$
    • $I$ random $L + L'$-bit blinding values $b_1, \ldots, b_I$
    • $I$ random $L + L'$-bit blinding values $c_1, \ldots, c_I$
    • $2 \cdot I \cdot (L + 1)$ random wire keys $\nu^0_{il}, \nu^1_{il}$ for $1 \leq i \leq I, 1 \leq l \leq L + 1$
1: Generate a random key $\kappa_i$ for each node $i$ in the branching program
2: for $i = 1$ to $I$ do
3:     Generate $2 \cdot (L + 1)$ random wire keys $\nu^0_{il}, \nu^1_{il}, l = 1, \ldots, L + 1$
4:     Generate random $(L + L')$-bit blinding value $b_i$ and $c_i$ but enforce the $L+1$-bit of $c_i$ to be 1; store $b_i, c_i, b'_i = b_i \bmod 2^{L+1}$, and $c''_i = c_i \mod 2^L$
5:     if $P_i$ is a classification node $\langle d_i\rangle$ then
6:         Compute $S_i = \{\text{“label”}, d_i\}_{\kappa_i}$ by encrypting $\{\text{“label”}, d_i\}$ with the random key $\kappa_i$.
7:     else if $P_i$ is a decision node $\langle W_i, a_i\rangle$ then
8:         Create a secure garbled circuit $C_i$ which performs the function COMPARE$(x, b'_i, c'_i, \mathcal{L}, \mathcal{R})$, where $\mathcal{L} = \langle\mathbf{L}(i), \kappa_{\mathbf{L}(i)}\rangle$ and $\mathcal{R} = \langle\mathbf{R}(i), \kappa_{\mathbf{R}(i)}\rangle$
9:         Use $\nu^0_{il}, \nu^1_{il} (1 \leq l \leq L+1)$ to encode the input wires of $C_i$.
10:    end if
11: end for
12: Randomly permute all but the first node in the branching program by changing their subscript $i$ to $i'$
13: return $T' = \langle\{S_1, \ldots, S_I\}, \kappa_1\rangle$
```

V. Security and Privacy Analysis

Our construction in Section IV can correctly compute the diagnostic results based on the genomic information and biomedical sensing data of the user and preserve the privacy of both the HS and the user under the threat model in Section III-B.

Correctness. For Algorithm 1, we show that $d = 0$ if and only if the genomic pattern $\mathbf{t}$ matches $\mathbf{s}$. We first note that by our definition, $\mathbf{t}$ matches $\mathbf{s}$ if and only if $t_{\text{pos}_m} = s_m, 1 \leq m \leq M$. Also, e_d is an encryption of $\sum_{m=1}^{M} r_m \cdot (t_{\text{pos}_m} - s_m)$. If $t_{\text{pos}_m} = s_m, 1 \leq m \leq M$, we have $e_d = [0]$ and $d == 0$.

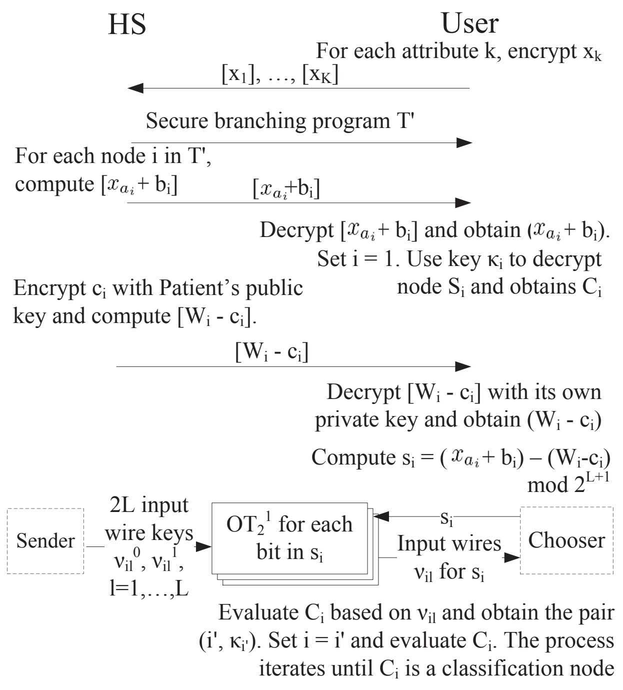
HS
User
For each attribute k, encrypt x_k
$[x_1], \ldots, [x_K]$
Secure branching program T'
For each node i in T', compute $[x_{a_i} + b_i]$
$[x_{a_i}+b_i]$
Decrypt $[x_{a_i} + b_i]$ and obtain $(x_{a_i} + b_i)$. Set i = 1. Use key κ_i to decrypt node S_i and obtains C_i
Encrypt c_i with Patient's public key and compute $[W_i - c_i]$.
$[W_i - c_i]$
Decrypt $[W_i - c_i]$ with its own private key and obtain $(W_i - c_i)$
Compute $s_i = (x_{a_i} + b_i) - (W_i - c_i) \mod 2^{L+1}$
Sender
2L input wire keys v_{il}^0, v_{il}^1, l=1,…,L
OT_2^1 for each bit in s_i
s_i
Input wires v_{il} for s_i
Chooser
Evaluate C_i based on v_{il} and obtain the pair $(i', \kappa_{i'})$. Set i = i' and evaluate C_i. The process iterates until C_i is a classification node

Fig. 2. Oblivious Evaluation of Genome-Aware Branching Program

Otherwise, there would be at least a random group element in e_d and thus the probability that e_d is an encryption of 0 is very low. Hence, if $e_d = [0]$, then $t_{\text{pos}_m} = s_m, 1 \leq m \leq M$ with overwhelming probability. The encrypted threshold $[W_i]$ can be further calculated based on the homomorphic property of the cryptosystem.

For the protocol in Fig. 2, we examine the evaluation process of the user to show that the user will reach a classification node of the branching program and his/her sequence to evaluate the branching program is based on the comparison result, i.e., go to the next evaluate node $L(i)$ if $x_{a_i} > W_i$ and to the next evaluate node $R(i)$ if $x_{a_i} < W_i$. Note that the user will first start with node 1 because he/she only has access to the secret key κ_1 of this node. If he/she successfully evaluates node 1, he/she will be able to obtain either $\mathcal{L} = \langle \mathbf{L}(1), \kappa_{\mathbf{L}(1)} \rangle$ or $\mathcal{R} = \langle \mathbf{R}(1), \kappa_{\mathbf{R}(1)} \rangle$. This enables him/her to decrypt either node $\mathbf{L}(1)$ with the associated secret key $\kappa_{\mathbf{L}(1)}$ or $\mathbf{R}(1)$ with $\kappa_{\mathbf{R}(1)}$ and further evaluate this node. The process will continue unless the next node is a classification node that contains a label instead of a garbled circuit, when the user will obtain the label assigned by the branching program based on his real-time sensing data **x**. To complete the proof of correctness, we only need to show that the user can obtain the correct output when he evaluates node $i, 1 \leq i \leq I$. The correctness of the garbled circuit has been proved in the literature [15], [16], [20], and we only use it as a blackbox. The input to the garbled circuit is $s_i = (x_{a_i} + b_i) - (W_i - c_i) \mod 2^{L+1}$, and the other parameters of the circuit $\beta = b'_i, \gamma = c''_i$. Note that γ should be an $L + 1$-bit integer so there is an extra bit 0 at the beginning of c''. With this input, the garbled circuit will compare $(x_{a_i} + b_i) - (W_i - c_i) - b'_i \mod 2^{L+1} = x_{a_i} - W_i + c'_i \mod 2^{L+1}$ and c'_i. Note that the result of comparing $x_{a_i} - W_i + c'_i \mod 2^{L+1}$ and c'_i is the same as the result of comparing $x_{a_i} - W_i$ and 0. In other words, the user will get $\mathcal{L}(i)$ if $x_{a_i} - W_i > 0$ and $\mathcal{R}(i)$ otherwise.

User Privacy. In Algorithm 1, it is easy to see that the SPU does not learn any additional information about the user because it only has access to genomic data encrypted under the private key of the user. As for the HS, it only performs operations over data encrypted by the additive homomorphic cryptosystem. The privacy of the encrypted genomic data $\mathbf{t}$ and the encrypted output W_i is thus guaranteed based on the indistinguishability of additive homomorphic cryptosystem.

For the secure evaluation protocol in Fig. 2, the HS receives the ciphertext $[x_{a_i}]$ which is encrypted under the public key of the user and seems to be randomly distributed. The HS also participates in OT_2^1 protocols as the sender, but it learns nothing due to the property of OT.

HS Privacy. Similarly, in Algorithm 1, the user does not learn additional information of the branching program because the thresholds w_1 and w_0 are never sent to the user. The user only has access to d and d', but since the user will receive several instantiations of d when computing $[W_i]$ for different nodes i, he/she will not know which attribute d represents. This information can be further hidden by introducing additional encrypted bits of 0 and random strings.

For the secure evaluation protocol in Fig. 2, the user receives T' from the HS. The user can only decrypt partial of the nodes in T' while the rest of the nodes are random ciphertexts to him/her. For the nodes that he/she can decrypt, say, node i, $W_i - c_i$ and $x_{a_i} + b_i$ is obtained, which statistically hides W_i and x_{a_i} when b_i and c_i are sufficiently large. Thus, the user will not know which attribute is being evaluated and what is the value of the threshold. The user also chooses the input wires for the garbled circuit C_i through OT_2^1, but the oblivious transfer protocol ensures that the user will not learn anything else and thus he/she will only obtain one of the two outputs of C_i and decrypt only one node at a time. The protocol only reveals the total number of nodes I and the length of the evaluation path in the branching program, which is acceptable in most cases in practice. However, if there is an upper bound of I, it can also be hidden by introducing some pseudo nodes. The length of the evaluation path can be also hidden by transforming the original branching program into a full decision tree at extra cost.

TABLE II
RUNNING TIME (IN UNIT OF SECONDS) OF THE OFFLINE MODULE

#attributes N (#nodes $I = 63$)	10	50	200	1000	5000
Estimated running time of HS	2	3	3	3	3
#nodes I (#attributes $K = 1000$)	15	31	63	127	255
Estimated running time of HS	1	2	2	4	4

VI. PERFORMANCE EVALUATION

We first evaluate the scalability of the proposed approach based on benchmarks provided in [16] and [21]. We use a PC with an Intel i7-3770 3.4GHz quad-core CPU to simulate the

TABLE III
RUNNING TIME (IN UNIT OF SECONDS) OF THE ONLINE MODULE

#attributes N (#nodes $I = 63$)	10	50	200	1000	5000
Estimated running time of HS	7	7	7	7	7
Estimated running time of user	13	14	15	20	43
#nodes I (#attributes $K = 1000$)	15	31	63	127	255
Estimated running time of HS	3	5	7	14	28
Estimated running time of user	10	12	14	27	51

computing capability of the user. The computing power of the HS can be much higher than that of the user in practice. Our protocols consist of an offline module for creating the secure genome-aware branching program and an online module for real-time health monitoring.

The offline module (Algorithm 1 and Algorithm 2) is performed collaboratively by the CI, SPU, and HS. The computation cost for the CI is to encrypt the genome of the user with an additive homomorphic cryptosystem, which takes around $2,580$ hours if the underlying homomorphic cryptosystem is the EC-ElGamal and 115 hours if the underlying homomorphic cryptosystem is AH-ElGamal. This is acceptable because the encryption only happens once and can be re-used by other genome-based applications as well. The SPU only stores and sends the encrypted data once to the HS and thus has low computation and communication overhead. We now analyze the scaling behavior of the HS for creating a secure genome-aware branching program. Compared to the previous encryption process, the scaling behavior for creating a secure branching program is more critical because it is a recurring process. According to the benchmark in [16], we expect the HS to have a nice scaling property. The estimated time for different numbers of nodes and different numbers of the attributes is given in Table II. The computation and bandwidth requirements of the HS are independent of the size of the attribute vector K, and will scale linearly with the number of nodes in the branching program. The offline module can be pre-computed in batch and could be parallelized using multiple servers. Therefore the aforementioned computation and communication overhead is acceptable in practice.

For the online module, both the user and the HS are involved. The computation cost for the HS has the same property with the offline module, i.e., the estimated time for the HS is independent of the number of the attributes K and will scale linearly with the number of nodes in the branching program. The user's computation time depends linearly on the number of nodes in the branching program. However, unlike the HS, the computation time of the user also depends on the size of the attribute vector, because the user has to encrypt his/her attributes before the evaluation. The expected time for the online module based on the nodes of the branching program and the number of the attributes is given in Table III. Generally speaking, our privacy-preserving protocols have acceptable computation and communication cost.

VII. CONCLUSION

In this paper, we have proposed a secure and privacy-preserving scheme for genome-aware remote health monitoring systems. The proposed protocols in the scheme enable the healthcare service provider to create a genome-aware branching program based on genomic data of a user without learning his/her sensitive genomic information, and also enable the user to securely evaluate the personalized program without revealing his/her sensitive biomedical sensing data to the healthcare service provider or learning the branching program. Our scheme has been shown to be secure and privacy-preserving, and has a good scaling property for the users.

REFERENCES

[1] "Advanced remote patient monitoring systems," Kalorama Information, Tech. Rep., July 2015. [Online]. Available: http://www.kaloramainformation.com/Advanced-Remote-Patient-9123949/

[2] E. Ayday, J. L. Raisaro, M. Laren, P. Jack, J. Fellay, and J.-P. Hubaux, "Privacy-preserving computation of disease risk by using genomic, clinical, and environmental data," in *Proceedings of USENIX Security Workshop on Health Information Technologies (HealthTech" 13)*, 2013.

[3] [Online]. Available: https://www.23andme.com/

[4] [Online]. Available: https://www.counsyl.com/

[5] H. Lin, J. Shao, C. Zhang, and Y. Fang, "Cam: cloud-assisted privacy preserving mobile health monitoring," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, pp. 985–997, 2013.

[6] L. Guo, Y. Fang, M. Li, and P. Li, "Verifiable privacy-preserving monitoring for cloud-assisted mhealth systems," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 1026–1034.

[7] G. Wang, R. Lu, and C. Huang, "Pguide: An efficient and privacy-preserving smartphone-based pre-clinical guidance scheme," in *GLOBECOM*. IEEE, 2015, pp. 1–6.

[8] H. Zhu, X. Liu, R. Lu, and H. Li, "Efficient and privacy-preserving online medical pre-diagnosis framework using nonlinear svm," *IEEE J. Biomed. Health. Inf.*, to appear.

[9] X. Liu, R. Lu, J. Ma, L. Chen, and B. Qin, "Privacy-preserving patient-centric clinical decision support system on naive bayesian classification," *IEEE J Biomed Health Inf*, vol. 20, no. 2, pp. 655–668, 2016.

[10] E. Ayday, J. L. Raisaro, J.-P. Hubaux, and J. Rougemont, "Protecting and evaluating genomic privacy in medical tests and personalized medicine," in *WPES*. ACM, 2013, pp. 95–106.

[11] N. Karvelas, A. Peter, S. Katzenbeisser, E. Tews, and K. Hamacher, "Privacy-preserving whole genome sequence processing through proxy-aided oram," in *WPES*. ACM, 2014, pp. 1–10.

[12] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*. Springer, 1999, pp. 223–238.

[13] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *CRYPTO*. Springer, 1984, pp. 10–18.

[14] O. Ugus, D. Westhoff, R. Laue, A. Shoufan, and S. A. Huss, "Optimized implementation of elliptic curve based additive homomorphic encryption for wireless sensor networks," *arXiv preprint arXiv:0903.3900*, 2009.

[15] A. Yao, "How to generate and exchange secrets," in *FOCS*. IEEE, 1986, pp. 162–167.

[16] J. Brickell, D. E. Porter, V. Shmatikov, and E. Witchel, "Privacy-preserving remote diagnostics," in *CCS*. ACM, 2007, pp. 498–507.

[17] I. F. Blake and V. Kolesnikov, "Strong conditional oblivious transfer and computing on intervals," in *ASIACRYPT*. Springer, 2004, pp. 515–529.

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

[19] B. Carlson, "Snps-a shortcut to personalized medicine," *Genetic Engineering & Biotechnology News*, vol. 28, no. 12, pp. 12–12, 2008.

[20] Y. Lindell and B. Pinkas, "A proof of security of yaos protocol for two-party computation," *Journal of Cryptology*, vol. 22, no. 2, pp. 161–188, 2009.

[21] E. De Cristofaro, S. Faber, and G. Tsudik, "Secure genomic testing with size-and position-hiding private substring matching," in *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*. ACM, 2013, pp. 107–118.