Verifiable Privacy-preserving Monitoring for Cloud-assisted mHealth Systems

Linke Guo*, Yuguang Fang†, Ming Li‡, and Pan Li§
*Department of Electrical and Computer Engineering, Binghamton University,
State University of New York, Binghamton, NY 13902, USA
†Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
‡Department of Computer Science and Engineering, University of Nevada, Reno, NV 89557, USA
§Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762, USA
Email: lguo@binghamton.edu, fang@ece.ufl.edu, mingli@unr.edu, li@ece.msstate.edu

***Abstract*—Widely deployed mHealth systems enable patients to efficiently collect, aggregate, and report their Personal Health Records (PHRs), and then lower the costs and shorten their response time. The increasing needs of PHR monitoring require the involvement of healthcare companies that provide monitoring programs for analyzing PHRs. Unfortunately, healthcare companies are lack of the computation, storage, and communication capability on supporting millions of patients. To tackle this problem, they seek for the help from the cloud. However, delegating monitoring programs to the cloud may incur serious security and privacy breaches because people have to provide their identity information and PHRs to the public domain. Even worse, the cloud may mistakenly return the incorrect computation results, which will put patients' life in jeopardy. In this paper, we propose a verifiable privacy-preserving monitoring scheme for cloud-assisted mHealth systems. Our scheme allows patients to verify the correctness of computation results from the cloud without revealing their PHRs and identity information. In addition, our advanced schemes offer efficient PHR updates and PHR computations on complex monitoring programs. By detailed performance evaluation, we have shown the security and efficiency of our proposed scheme.**

***Index Terms*—PHR, Privacy, Verifiable Computation, mHealth**

I. INTRODUCTION

It has been witnessed that the explosive growth of mobile devices, such as smartphones, tablets, and mobile sensors, has changed the ways of people's living. Particularly, mHealth (mobile health), which leverages widely deployed remote medical devices to offer accurate, timely, and low-cost medical services, has already shown the great potential on improving the quality of healthcare services as well as the quality of life (QoL). As an evidence of its long-term impacts, both Apple and Google have announced their initiatives on healthcare recently. The launched services, Apple HealthKit and Google Fit, use remote sensor-associated mobile devices to *provide a hub for users to keep track of health related information in the following decade* [1]. Patients can periodically collect, aggregate, and deliver the readings on medical sensors to a service provider, and then retrieve back the analyzed results. For example, Microsoft has launched project "MediNet" in Caribbean countries [2], which tries to keep monitoring the health status of patients with diabetes and cardiovascular diseases. Patients use wireless body sensor networks (WBSNs) to collect and report various types of physiological data, such as blood pressure, breathing rate, blood glucose level, electrocardiogram, and peripheral oxygen saturation. Then, patients could get various medical consultation like physical activity assistants, insulin intake, and cardiac analysis, from service providers that focus on developing the corresponding healthcare monitoring programs.

This work was partially supported by US National science Foundation under grant CNS-1423165 and CNS-1343356. The work of Pan was partially supported by US National science Foundation under grant CNS-1343220 and CNS-1149786.

Unfortunately, the above promising features and thriving economical initiatives may not succeed unless the following issues have been well addressed. On the one hand, for the business confidentiality concerns, healthcare companies may not reveal their monitoring program details as embedded apps to every user. On the other hand, some small startup companies may be able to help store and compute for a small number of patients. However, they may face the limitation on their computation, storage, and communication capabilities when the number of users increases. Inspired by [3], the proposed architecture enables the outsourcing of the computational intensive part to the cloud service provider, which helps patients greatly reduce the computation loads on decrypting the encrypted PHRs. Hence, a viable solution would be seeking the help from cloud service providers, by which companies can delegate monitoring programs and outsource the computation load. However, this promising approach incurs obvious security and privacy breaches. First, patients' identity information should be bound with the PHRs for accurate diagnosis, but existing approaches fail to preserve the identity privacy when the patient provides her PHRs to the cloud. Second, required by HIPPA (Health Insurance Portability and Accountability Act), all patients' PHR should be stored in a ciphertext form. Patients may not want to disclose their PHRs to the cloud due to privacy concern, in the sense that computation outsourcing may potentially expose patients' PHRs under current situation. Third, as reported in [4], the cloud service provider may experience hardware/software failures, human errors and external malicious attacks, which may bring life-threatened issue to patients if the retrieved results are not correctly computed. To address the above issues, we propose the verifiable privacy-preserving monitoring for cloud-assisted mHealth systems. Our scheme preserves the privacy of patients' identity information during the verification, and also allows patients to verify the correctness of outsourced computation results, while maintaining the privacy of patients' PHRs to the cloud for computation.

Related Works:

Privacy-preserving Identity Verification: To deal with the potential risks of privacy exposure, several eHealth systems [5]–[9] enable patients to encrypt their PHRs before storing it on the central storage. Although the encrypted PHRs prohibit the centralized facility from obtaining private information, it still faces identity leakage when patients retrieve their PHRs. Another privacy leakage is regarding the verifiability on patients' real identities. For the system with a centralized infrastructure, the verification cloud be easily managed. However, for a cloud-assisted architecture, patients have to show their real identities and the corresponding PHRs to the could service provider in order to get verified, which bring the possibility of identity leakage and impersonation attacks to patients. To tackle this problem, we apply the non-interactive zero-knowledge proof system [10]–[12] and techniques in [13], [14], which simultaneously achieve the privacy preservation and the verifiability of the private identity information.

Verifiable Computation: The verifiable computation is first introduced in [15], which tries to build up the framework for offloading the computation of some functions to other untrusted clients, while maintaining verifiable results. Parno *et al.* [16] discuss the delegation of verification in public domain, which shares the similar idea with our proposed scheme. In [17]–[19], they formally define the definition of publicly verifiable computation (PVC). However, the above schemes only support the plaintext computation, which means for users, they have to upload plaintext to the untrusted cloud and expect the computational results. As an extension of PVC, Papamanthou *et al.* in [20] introduce the signature of correct computation, which uses multi-polynomials for verification. Inspired by this work and its previous work [21], we add the privacy-preserving feature in PVC to fulfill the requirements in mHealth monitoring scenarios.

Privacy-preserving Mobile Sensing: Another line for this research is the privacy-preserving mobile sensing [22]–[24], which exploits data contributed by mobile users to monitor and evaluate an environment. However, their works mainly concern the privacy leakage on the aggregated data as well as the incentives without considering the impact of the integrity of the aggregation results.

Our Contributions:

- We propose a privacy-preserving identity verification scheme to verify patients' private identity information to the semi-trusted cloud service provider.
- Different from current PVC schemes, we provide the privacy of reported PHRs while still maintaining the capability of computation on PHRs.
- Our scheme enable patients to check the correctness of outsourced computation results.
- The healthcare monitoring company does not need to reveal its programs to every individual user.
- We extend our basic scheme with two advanced schemes for efficient update and supporting multi-variable monitoring programs.

The remainder of this paper is organized as follows. Section II introduces preliminaries and assumptions. We describe the system model in Section III, along with the design objectives. The proposed scheme is presented in detail in Section IV, followed by the protocol evaluation in Section V. Finally, Section VI concludes the paper.

II. Preliminaries and Problem Formulation

A. Pairing

A pairing function (simply *pairing*) is a map $e : G \times G \rightarrow G_1$, where G and G_1 are two multiplicative cyclic groups of the same prime order n with the following properties [25], [26].

- **Bilinearity:** For all $g, h \in G$ and random numbers $a, b \in Z_n^*$, it has $e(g^a, h^b) = e(g, h)^{ab}$.
- **Computability:** There exists a computable algorithm that can compute e efficiently.
- **Non-degeneracy:** For $g \in G$, $e(g, g) \neq 1$.

B. Cryptographic Assumptions

Definition 1. ***Discrete Logarithmic Problem (DLP)*** *[27]*

Let g, h be two elements in G. It is computationally intractable to find an integer a, such that $h = g^a$.

Definition 2. ***Subgroup Decision Problem (SDP)*** *[28]*

Let $n = pq$ be the order of G. Given (n, G, G_1, e) and $x \in_R G$, it is computationally intractable to decide whether x is the element in G_p or G_q when the factorization of n is unknown.

Definition 3. ***Computational Diffie-Hellman (CDH) Problem*** *[29]*

Given (g, g^a, g^b) for given random numbers a, b, it is computationally intractable to compute the value of g^{ab}, where G is a cyclic group of order n and g is a generator of G.

C. Problem Formulation

As an illustration, we first present our problem formulation. The basic idea of our verifiable privacy-preserving PHR computation is to allow patients to verify the correctness of returned computation results, while maintaining patients' privacy on their input data. Many healthcare monitoring programs are developed via the regression analysis based on experimental results [30], and then apply some curve fitting approaches, e.g., Least Square Fitting (LSF), to mathematically find the best fitting curve. Particularly to mHealth monitoring, many curves in the regression analysis turns out to be polynomials [31], including single-variable polynomial and multi-variable polynomial. As an example, in pharmaceutical, the responses for daily dietary supplements based on days treated [32] could be fitted to the following single-variable polynomial,

$$y = \alpha + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + e$$

where x is the monitored intake and y is the response level. The multi-variable polynomial case can be found in [33], which contains more than 50 terms on 3 variables.

Our scheme relies on the following algebraic results.

Lemma 1 (Polynomial Decomposition): Let $f(x)$ be a polynomial. For all $m \in Z_n$, there exists polynomial $w(x)$ (which can be found in polynomial-time) such that polynomial $f(x) - f(m)$ can be expressed as $f(x) - f(m) \equiv (x - m)w(x)$.

Lemma 2 (n-Independent Polynomials Decomposition): Let $f_i(x_i)$ be a polynomial, and $F(\vec{x}) = \sum_{i=1}^{z} f_i(x_i)$, where $\vec{x} := \{x_i\}_{i=1}^{z}$. For all $\vec{m} \in Z_n$, there exists polynomials $w_i(x_i)$ (which can be found in polynomial-time) such that each polynomial $f_i(x_i) - f_i(m_i)$ can be expressed as $f_i(x_i) - f_i(m_i) \equiv (x_i - m_i)w_i(x_i)$, which gives $F(\vec{x}) - F(\vec{m}) \equiv \sum_{i=1}^{z}(x_i - m_i)w_i(x_i)$.

Our scheme leverages the above properties to take $\vec{m}$ as the input of PHR raw data, and sends the encrypted form of $\vec{m}$ to the cloud service provider to expect the results $F(\vec{m})$. While performing the computation on $\vec{m}$, the cloud outputs a provable signature, which shows it indeed computes $F(\vec{m})$. Since the patient is the only one who knows $\vec{m}$, she can validate the correctness of the results by checking the equality of identity functions in Lemma 1 and Lemma 2 after decrypting the ciphertext form of $F(\vec{m})$.

III. System Model

A. Network Model

We first give a brief introduction to our proposed cloud-assisted mHealth system. As shown in Fig. 1, the system mainly consists of four entities, trust authority (TA), health-care company, cloud service provider, and users with mobile devices and the corresponding sensing devices. We list our assumptions and functionalities for these entities below,

- Trust Authority (TA): The fully-trusted TA is responsible for the system security setup. The security setup includes distributing public parameters, issuing secret information, and system maintenance. TA may go offline after the system starts unless new users come in.
- Cloud Service Provider: The cloud service provider (cloud for short) is considered to be a semi-trusted infrastructure, which is curious but not malicious.
- Healthcare Company: We refer to a healthcare company as "*company*" in the description of our design. The company has limited computational resources to provide PHR computation for millions of patients, but it is able to delegate the corresponding health programs, like polynomial/multi-polynomial functions, to the cloud.
- Users: Patients use their mobile devices to collect and report PHRs to the cloud, then retrieve computation results. They are able to verify the correctness of the results.

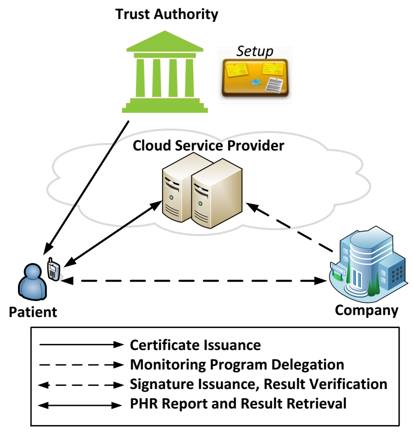

Trust Authority
Setup
Cloud Service Provider
Patient
Company
Certificate Issuance
Monitoring Program Delegation
Signature Issuance, Result Verification
PHR Report and Result Retrieval

Fig. 1. System Model

B. Design Objectives

We have two main design objectives for the verifiable privacy-preserving monitoring scenario. First, patients should be able to verify the correctness of returned computation results. Then, patients' PHRs should be encrypted before sending to the cloud, and the patients private identity information should be kept undisclosed during the authorization and verification process. With this being said, the cloud cannot learn either the raw data reported by patients or the relationship between identity information and the computation results. We also consider the confidentiality of the monitoring program, which should be kept undisclosed to patients. Otherwise, malicious users may obtain the program detail and anyone would be able to use it for free.

C. Adversarial Model

For the fully-trusted trust authority, we assume that it is uncompromisable by any kind of adversaries during the scheme run. The company is assumed to be semi-trusted, and it may launch inference attack on on patients' identities by linking the queried programs and patients' identity information. For the cloud, it is assumed curious but honest, in the sense that it will follow the proposed scheme operation on computing the reported PHRs, but may analyze the statistic of encrypted raw data. Most importantly, the massive operations on the cloud side may incur incorrect computation results, which leads to severe security breaches on the integrity of PHRs. Malicious users may try to observe/intercept heath monitoring programs.

We exclude several types of attacks that are beyond the discussion of this paper. The delegation of monitoring programs to the cloud is not considered as attacks on compromising the confidentiality of the company, because revealing the programs will not offer benefit to the cloud and can be easily traced for punishment. Our monitoring scheme may fail to fight against the denial-of-service attacks when numbers of malicious patients send requests to the cloud with dummy PHRs. Collusion attacks may not be thwarted when malicious users use brute force to compromise encrypted PHRs. The insider attack and global observer attack are also not considered in this paper.

IV. Our Proposed Scheme

In this section, we present our cloud-assisted verifiable privacy-preserving monitoring scheme in detail, which mainly consists of two main building blocks, privacy-preserving identity verification and verifiable PHR computation. We will first discuss the single variable polynomial verifiable PHR computation, and then give the advanced scheme on efficient update and multi-variable polynomial case.

A. Overview

In our scheme, a patient, Alice, first obtains a private parameter σ denoting the unique verifiable identity from TA. Then, based on her request, the company issues a blind signature Ψ on σ showing that Alice has registered the corresponding monitoring program $f(\vec{x})$, where $f(\vec{x})$ is a single/multi-polynomial function. Then, Alice starts to periodically collects her medical data via sensing devices and stored as PHR as $\vec{x} = (x_1, x_2, ..., x_n), x_i \in Z_q^*$, where each entry of the vector denotes unique monitored data, like heart rate, blood pressure, blood glucose level, etc. Note that $\vec{x}$ could be single entry vector representing single variable monitoring program. To request for computation results on $\vec{x}$, Alice first encrypts the vector as $\vec{c} := E(\vec{m})$, where $\vec{m}$ is the monitored raw data (the input of $f(\vec{x})$) and $E(\cdot)$ is the encryption scheme, then sends it together with Ψ to the cloud. Meanwhile, Alice generates a set of zero-knowledge proofs on $\vec{c}$ and Ψ for privacy-preserving verification. If the cloud could publicly verify the validity of the blind signature Ψ, it will compute the program function $f(\vec{x})$ on the input $\vec{c}$. Finally, the cloud outputs the computation results $f(E(\vec{m}))$ as well as a provable signature δ. Alice can decrypted and accept the results if and only if she can verify the correctness of $f(\vec{m})$ and δ based on the plaintext form of PHR $\vec{m}$.

B. System Setup

TA first generates a set of parameters for the system, and publishes corresponding public parameters for patients to create their proofs and encrypted PHRs, as well as for the cloud to generate the provable signature. TA may go offline after the successful setup. For the ease of description, we list the system setup as follows for different objectives.

1) General Setup: Given a security parameter ξ, TA outputs a tuple $param := (n, G, G_1, e)$ to the public domain, where $n = pq$ is the order of G and p, q are large primes.

2) Partially Blind Signature Setup: Along with the above settings, we assume the factorization of n is hard and subgroup decision problem is hard in both G and G_1. In addition to the above tuple, TA outputs the domain public parameter $(g, g^s) \in G^2$, where $s \in_R Z_n^*$ is chosen as master secret key.To generate the signature, TA defines two hash functions $H : \{0,1\}^* \rightarrow G$ and $H_0 : \{0,1\}^* \rightarrow Z_n^*$. Then, TA issues the signing key pair pk/sk to the company, where $pk := H(id_c) \in G$ and $sk := H(id_c)^s$.

3) Proving and Monitoring System Setup: TA randomly chooses another random generator $g_0 \in G$ and set $h = g_0^p \in_R G_q$, and publishes h together with $param$ as common reference string crs. For each valid patient, TA issues a private certificate $\sigma = g^{\frac{1}{s+id_A}}$ using Boneh-Boyen signature scheme [34] given Alice's ID id_A, which can be used for verification together with public information g^s. Given the monitoring public key $\texttt{pk} := (n, G, G_1, e, g, h)$, TA also issues Alice the private key $\texttt{sk} := q$ for decrypting the computation results.

C. Privacy-preserving Identity Verification

Our privacy-preserving identity verification scheme is constructed and modified based on the partially blind signature scheme in [35] and zero-knowledge proof system in [36]. To guarantee both the privacy and verifiability on σ, Alice first sends a request to the company and asks for a blind signature on σ. Then, Alice sends the zero-knowledge proof of σ and the corresponding parameters to the cloud for further verification. Our proposed scheme consists of four subprotocols, `Request`, `BlindSign`, `ProveGen`, and `Verify`.

1) Signature Request: $(\theta, \phi) \leftarrow \texttt{Request}(g, pk, id_A, w)$: Alice first requests for some parameters from the company for partially blind signature on σ. Before the request is sent, the company and Alice have to agree on certain messages, which will be embedded in the signature. This negotiation process could be easily achieved by either downloading the app with certain additional notified information or confirming agreement (receipt) on both sides. Given the registered ID id_A, both Alice and the company agree on a string $l \in \{0,1\}^n$. Then, the company randomly chooses $t \in Z_n^*$, computes $\theta = g^t, \phi = pk^t = H(id_c)^t$, and sends (θ, ϕ) back to Alice.

2) Partially Blind Signature Generation Process: $\epsilon' \leftarrow \texttt{BlindSign}(\theta, g^s, \phi, l, \sigma)$: To generate the signature, Alice first chooses $\alpha, \beta, \gamma \in_R Z_n^*$, and then computes $\theta' = \theta^\alpha \cdot (g^s)^\gamma = g^{\alpha t + \gamma s}$, $\phi' = H(id_c)^{\alpha(\beta+t)} H(l)^{-\gamma}$, and $u = \alpha^{-1} H_0(\sigma || \phi') + \beta$. After receiving (θ', ϕ', u), the company computes $\epsilon = H(id_c)^{s(t+u)} H(l)^t$. Then, Alice unblinds the received ϵ by computing $\epsilon' = \epsilon^\alpha$. Different from the original scheme in [35], we embed the validity check on σ during the verification of ϵ'.

3) Commitment and Proof Generation Process: $(\mathsf{com}_i, \pi) \leftarrow \texttt{ProveGen}(\theta', \phi', \epsilon', \sigma, l)$. This process happens on Alice's side. First of all, we assume the validity of $H_0(\sigma || \phi')$, in the sense that Alice will not use fabricated σ during the process of generating commitment and proofs used to prove to the cloud. Otherwise, the company can use log to trace back the malicious use together with the cloud. Then, we assume the cloud could not identify the correlation between the string l and the identity of patients (e.g., using 1 string for all users who register to the same company). Namely, we want the cloud to test the equality of the following equation, so that it can successfully verify the identity of Alice by giving the partially blind signature $\Psi := (\theta', \phi', \epsilon', \sigma, w)$.

$$e(\epsilon', g) e(g^{id_A} g^s, \sigma) \stackrel{?}{=} e(g \cdot \phi' \cdot H(id_c)^{H_0(\sigma || \phi')}, g^s) e(H(l), \theta') \tag{1}$$

However, directly revealing the given set and identity information id_A to the cloud will not only disclose the verification information, but also reveal the correlation of ID and partially blind signature ϵ'. Instead, we try to apply the approach in [36] to achieve the zero-knowledge proof in verifying the correctness of the signature as well as the certificate. First, we rewrite Eq.1 as follows,

$$e(\epsilon', g) e(g^{id_A} g^s, \sigma) e(\phi' \cdot H(id_c)^{H_0(\sigma || \phi')}, g^{-s}) e(H(l)^{-1}, \theta') \stackrel{?}{=} e(g, g). \tag{2}$$

Then, Alice first chooses random numbers $\mu_i, \nu_i \in_R Z_n, i = 1, 2, 3, 4$, to generate the corresponding commitments.

$$\mathsf{com}_1 := \epsilon' h^{\mu_1} = H(id_c)^{\alpha s(t+u)} H(l)^{\alpha t} h^{\mu_1}, \mathsf{com}_1' := g h^{\nu_1}$$

$$\mathsf{com}_2 := g^{id_A + s} h^{\mu_2}, \mathsf{com}_2' := \sigma h^{\nu_2} = g^{\frac{1}{s+id_A}} h^{\nu_2}$$

$$\mathsf{com}_3 := \phi' \cdot H(id_c)^{H_0(\sigma || \phi')} h^{\mu_3}, \mathsf{com}_3' := g^{-s} h^{\nu_3}$$

$$\mathsf{com}_4 := H(l)^{-1} h^{\mu_4}, \mathsf{com}_4' := \theta' h^{\nu_4} = g^{\alpha t + \gamma s} h^{\nu_4}$$

Based on the commitment set, Alice constructs the proof $\pi := \prod_1^4 (\mathsf{com}_i h^{-\mu_i})^{\nu_i} (\mathsf{com}_i')^{\mu_i}$. Then, Alice sends $(\{\mathsf{com}_i, \mathsf{com}_i'\}_{i=1}^4, \pi)$ to the cloud for verification.

4) Privacy-preserving Identity Verification Process: $(0, 1) \leftarrow \texttt{Verify}(\{\mathsf{com}_i, \mathsf{com}_i'\}_{i=1}^4, \pi, h, e(g, g))$. Given the above parameters, the cloud will verify the equality of the following equation, and output 1 for successful verification, 0 as otherwise, respectively.

$$\prod_{i=1}^4 e(\mathsf{com}_i, \mathsf{com}_i') = e(g, g) e(\pi, h) \tag{3}$$

Proof. **Correctness:** We prove the correctness of our construction based on Eq. 3.

$$
\begin{aligned}
&\prod_{i=1}^{4} e(\mathsf{com}_i, \mathsf{com}'_i)\\
&= e(\epsilon' h^{\mu_1}, g h^{\nu_1}) e(\phi' \cdot H(id_c)^{H_0(\sigma||\phi')} h^{\mu_3}, g^{-s} h^{\nu_3})\\
&\cdot e(H(l)^{-1} h^{\mu_4}, g^{\alpha t+\gamma s} h^{\nu_4}) e(g^{id_A+s} h^{\mu_2}, g^{\frac{1}{s+id_A}} h^{\nu_2})\\
&= e(H(id_c)^{\alpha s(t+u)} H(l)^{\alpha t}, g) e(\phi' \cdot H(id_c)^{H_0(\sigma||\phi')}, g^{-s})\\
&\cdot e(H(l)^{-1}, g^{\alpha t+\gamma s}) e(h^{\mu_1}, g) e(\epsilon' h^{\mu_1}, h^{\nu_1}) e(h^{\mu_3}, g^{-s})\\
&\cdot e(\phi' \cdot H(id_c)^{H_0(\sigma||\phi')} h^{\mu_3}, h^{\nu_3}) e(h^{\mu_4}, g^{\alpha t+\gamma s})\\
&\cdot e(H(l)^{-1} h^{\mu_4}, h^{\nu_4}) e(g^{id_A+s} h^{\mu_2}, g^{\frac{1}{s+id_A}} h^{\nu_2})\\
&= e(H(id_c)^{\alpha s(t+u)}, g) e(H(id_c)^{\alpha(\beta+t)+H_0(\sigma||\phi')}, g^{-s})\\
&\cdot e(H(l)^{\alpha t}, g) e(H(l), g)^{\gamma s-\alpha t-\gamma s} e(g^{\mu_1}, h) e((\epsilon' h^{\mu_1})^{\nu_1}, h)\\
&\cdot e(g^{-s\mu_3}, h) e((\phi' \cdot H(id_c)^{H_0(\sigma||\phi')} h^{\mu_3})^{\nu_3}, h) e(g^{\mu_4(\alpha t+\gamma s)}, h)\\
&\cdot e((H(l)^{-1} h^{\mu_4})^{\nu_4}, h) e(g, g) e(g^{(s+id_A)\nu_2+\frac{\mu_2}{s+id_A}}, h) e(h^{\mu_2\nu_2}, h)\\
&= e(g, g) \prod_{i=1}^{4} e((\mathsf{com}_i h^{-\mu_i})^{\nu_i} (\mathsf{com}'_i)^{\mu_i}, h) = e(g, g) e(\pi, h)
\end{aligned}
$$

□

The verification outputs 1 if Alice provides the commitments and proofs on correct σ and Ψ. It is clear to see that we hide the partially blind signature as well as Alice's private identity information during the verification to the cloud.

D. Verifiable PHR Computation

As long as Alice passes the verification on her identity, the cloud can further process Alice's uploaded PHRs. We will first discuss the situation where the single-variable polynomial program with update, and then go through the multi-variable polynomial monitoring program.

1) Monitoring Program Delegation: This process enables the company to delegate program to the cloud, and allows the cloud to compute on Alice's input PHR. Taking Alice's blood pressure testing program as an example, the coefficients of the single-variable k-degree polynomial can be represented as a vector $\vec{a} := (a_0, a_1, ..., a_k), a_i \in Z_n$ denoting the following monitoring function,

$$f(x) = \sum_{i=0}^{k} a_i x^i = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_1 x + a_0.$$

To delegate the above function to the cloud, the company sends the coefficient vector $\vec{a}$ to the cloud along with the agreement string l for identifying the corresponding program. Note that the delegation of the program is not considered as loosing the confidentiality of monitoring programs, because it does not offer the cloud business benefit to disclose the confidential information of a company in this design.

2) PHR Encryption: As we point out in the introduction part, the PHR should be encrypted before sent to the cloud for computation. We extend the PVC scheme in [20] and the BGN encryption scheme in [28] as our basic cryptographic tools. Note that the reported PHR m could be one entry among a time-serie based data vector $\vec{m} := (m_1, m_2, ...m_n), m_i \in Z_n$. Depending on the degree of the delegated polynomial function, Alice first randomly chooses a set of numbers $\vec{r} := (r_0, r_1, ...r_k)$, where $r_i \in Z_n$, and sends $\vec{r}$ to the company. Upon receiving $\vec{r}$, the company computes $\vec{r'} := \vec{r} \cdot \vec{a} = (a_0 r_0, a_1 r_1, ..., a_k r_k)$, and sends $h^{\bar{r}} = h^{\sum_{i=0}^{k} r'_i}$ and $g^{\bar{r}}$ to Alice. Here, the company will also send $\bar{r}$ to the cloud for the computation need. Note that the cloud would not be able to identify each r_i from $\bar{r}$ even if the cloud knows $\vec{a}$ due to the linear independence. Then, Alice generates the ciphertext of PHR c by selecting another random number $d \in Z_n$ as follows,

$$c := \{g h^{d \cdot r_0}, g^{m} h^{d \cdot r_1}, g^{m^2} h^{d \cdot r_2}, ..., g^{m^k} h^{d \cdot r_k}\},$$

where each entry in c is computed as $c_i := g^{m^i} \cdot (h^{r_i})^d$.

To obtain the proof on the computation results on m, Alice sends $\{c, \lambda, H(l)\}$ to the cloud, where $\lambda = \frac{1}{(x-m) \cdot d} \bmod n$, and then requests the cloud to compute the monitoring program based on her input. We denote $x \in_R Z_n$ as a random point used in the polynomial function in the subsequent development. Meanwhile, Alice requests the company to generate a public parameter $g^{f(x)}$, which will be sent to the cloud for the PHR computation.

3) Verifiable PHR Computation: When the cloud receives the computation set, it recalls the corresponding monitoring program $\vec{a}$ and $\bar{r}$ based on $H(l)$. Then, the cloud computes the PHR as follows,

$$
\begin{aligned}
v &= \prod_{i=0}^{k} \left(g^{m^i} \cdot (h^{r_i})^d\right)^{-a_i}\\
&= \prod_{i=0}^{k} g^{-a_i \cdot m^i} \cdot h^{-a_i r_i d} = g^{\sum_{i=0}^{k} -a_i \cdot m^i} \cdot h^{\sum_{i=0}^{k} -a_i r_i d}\\
&= g^{-f(m)} \cdot h^{-d \sum_{i=0}^{k} r'_i}.
\end{aligned}
$$

To obtain the correct computation results as well as the provable signature δ, the cloud first computes $\lambda' := \frac{\lambda}{\bar{r}} = \frac{1}{(x-m) \cdot d \cdot \bar{r}}$, and it further computes the provable signature δ by using the public parameter $g^{f(x)}$,

$$
\begin{aligned}
\delta &= (g^{f(x)} \cdot v)^{\lambda'} = (g^{f(x)-f(m)} \cdot h^{-d \sum_{i=0}^{k} r'_i})^{\frac{1}{(x-m) \cdot d \cdot \bar{r}}}\\
&= g^{\frac{f(x)-f(m)}{(x-m)} \cdot \frac{1}{d\bar{r}}} \cdot h^{-\frac{1}{(x-m)}} = \left(g^{w(x)} \cdot h^{-\frac{d\bar{r}}{(x-m)}}\right)^{\frac{1}{d\bar{r}}},
\end{aligned}
$$

where $w(x)$ is a $(k-1)$-degree polynomial function satisfying $w(x) \equiv \frac{f(x)-f(m)}{(x-m)}$ if and only if $f(m)$ is the value based on the input of m.

Finally, the cloud sends $\{v, \delta\}$ back to Alice and waits for Alice's verification results on $f(m)$.

4) PHR Result Decryption and Verification: Upon receiving $\{v, \delta\}$, Alice can decrypt the ciphertext of v and obtain $f(m)$ by using the private key $\mathtt{sk} := q$,

$$\left(\frac{1}{v}\right)^q = (g^{f(m)} h^{d\bar{r}})^q = (g^q)^{f(m)} h^{d\bar{r}q} = (g^q)^{f(m)} \in G_p$$

To recover $f(m)$, it suffices to compute discrete log of $\frac{1}{v}$ with base g^q. Since we assume the limited integer space as $0 < m, f(m) < M$, we could obtain the final results using Pollard's lambda method [37] within $O(\sqrt{M})$. Note that this approach will not compromise the security level of the encryption algorithm on c, as the adversary fails to decrypt c

using the discrete log due to the much larger message space by using random numbers d and r_i.

Then, Alice also wants the proof on the decryption results $f(m)$, so she uses pseudonyms and SSL to send encrypted form of $(x, f(m))$-tuple to the company which helps her construct the coefficient vector on $w(x)$ as $\vec{w} := (w_0, w_1, ...w_{k-1})$, as well as the proving vector $W := g^{\sum_{i=0}^{k-1} w_i x^i}$ based on the point x. Alice also computes $(g^{\bar{r}})^d = g^{d\bar{r}}$ and $\eta = (h^{\bar{r}})^{-d/(x-m)}$. Then, she checks the equality of following equation to see whether the cloud correctly helps her compute the corresponding monitoring program,

$$e(W \cdot \eta, g) \overset{?}{=} e(\delta, g^{d\bar{r}}) \tag{4}$$

Proof. **Correctness:** It is easy to prove the correctness of the above equation.

$$\begin{aligned}\texttt{LHS} &= e(g^{w(x)}, g)e(h^{-\frac{d\bar{r}}{x-m}}, g)\\ \texttt{RHS} &= e(g^{\frac{f(x)-f(m)}{x-m}}, g)e(h^{-\frac{d\bar{r}}{x-m}}, g)\end{aligned}$$

We have shown the equality of both sides based on the assumption that $f(x) - f(m) = (x - m)w(x)$. □

E. Advanced Scheme for Verifiable PHR Computation

In this subsection, we discuss two extensions for our proposed verifiable PHR computation scheme, which better adapt different scenarios in mHealth monitoring.

1) Verifiable PHR Computation with Efficient Updates: Our basic scheme mainly discusses on the monitoring scenario that requires static single variable. However, for most healthcare monitoring programs, e.g., continuing glucose monitoring (CGM) systems with fingersticks 3-4 times per day for optimal glucose sensor accuracy, require multiple updates on one single variable. Hence, it is desirable to extend our basic scheme with efficient PHR updates in order to reduce the computational cost on the patient side.

To enable the cloud to compute a new result $f(m')$ on updated PHR m' using the basic scheme, the patient may need to compute a new set of c costing $2k$ exponentiation operations on G and G_q and k multiplication operations on G. Moreover, other required intermediate results also have to be completely updated, which brings massive computational loads to both the cloud and patients.

In our advanced scheme for efficient updates, we consider a set of PHR $\vec{m} := (m_1, m_2, ...m_n), m_i \in Z_n$, and the previous input data item m_i and ciphertext form c_i have been correctly computed as $f(m_i)$ given $\{v_i, \delta_i\}$. Then, with the updated PHR m_{i+1}, Alice firsts updates the random numbers by choosing $\vec{r}_{i+1} := \{r_{i+1,j}\}_{j=0}^{k}$ on Z_n, and obtains $g^{\bar{r}_{i+1}}$ and $h^{\bar{r}_{i+1}}$ from the company, as $\bar{r}_{i+1} = \sum_{j=0}^{k} a_j \cdot r_{i+1,j}$. Instead of generating a new set of c_{i+1}, Alice sends the update set $c'_{i+1} := \{g, g^{m_{i+1}-m_i}, g^{m_{i+1}^2-m_i^2}, ..., g^{m_{i+1}^k-m_i^k}\}$ and $\vec{\varrho}_{i+1} := \{d_{i+1}r_{i+1,j} - d_i r_{i,j}\}_{j=0}^{k}$, where $d_{i+1} \in_R Z_n^*$ is a random number selected by Alice for computing $f(m_{i+1})$. Upon receiving the above updated parameters, the cloud would be able to update the original c_i to c_{i+1} as follows,

$$c_{i+1,j} = c_{i,j} \cdot c'_{i+1,j} \cdot h^{\varrho_{i+1,j}} = g^{m_{i+1}^j} h^{d_{i+1} r_{i+1,j}}$$

which directly reduces the cost from (2k-exp+k-mul) to (k-exp) on Alice's side. Accordingly, Alice can update $\lambda_{i+1} = \frac{1}{(x-m_{i+1})\cdot d_{i+1}}$ and $h^{\bar{r}_{i+1} d_{i+1}}$ to the cloud. Then, the cloud could continue with the original scheme and output $\{v_{i+1}, \delta_{i+1}\}$ for Alice to decrypt and verify.

2) Multi-variable Polynomial Monitoring: For some cases, multiple factors would lead to a single output of some mHealth monitoring programs, in which we have to consider the verifiable PHR computation on multi-variable polynomial functions. Due to page limit, we will only focus on the scenario where the cross terms (like x_1x_2 and $x_3^2x_4^3$) does not appear in the multi-variable monitoring program.

To better address the above problem, we continue to use Alice's mHealth monitoring as an example, in which the monitoring program takes multiple factors, like *blood pressure*, *heart rate*, *glucose level*, etc, as inputs, and expects the output on evaluating her current health condition and drug usage. The z-variable mHealth monitoring program is denoted as follows,

$$\begin{aligned}F(\vec{x}) &= f_1(x_1) + f_2(x_2) + \cdots + f_z(x_z)\\ \text{where } f_1(x_1) &= a_{k_1}x_1^{k_1} + a_{k_1-1}x_1^{k_1-1} + \cdots + a_1x_1 + a_0\\ f_2(x_2) &= b_{k_2}x_2^{k_2} + b_{k_2-1}x_2^{k_2-1} + \cdots + b_1x_2 + b_0\\ &\vdots\\ f_z(x_z) &= z_{k_z}x_z^{k_z} + z_{k_z-1}x_z^{k_z-1} + \cdots + z_1x_z + z_0\end{aligned}$$

where k_i is the highest degree of each polynomial $f_i(x_i)$, and each entry $x_i \in \vec{x}$ represents different input PHRs from different monitoring devices used in the program. As we can see from our basic scheme, the trivial solution would be expecting each output of $f_i(x_i)$, and get verified one by one. Obviously, it takes lots of computation load, which would be $O(z \cdot \max\{k_i\})$ on the cloud side and $O(z)$ on the verification on the patient side. As our advanced scheme, we allow the cloud to aggregate the computation values on each $f_i(x_i)$, then it can generate one provable signature based on the aggregated result for verification.

The input PHR is considered to be a vector $\vec{\mathbf{m}} : \{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_z\}, \mathbf{m}_i \in Z_n$. Similar to our basic scheme, Alice can generate $\mathbf{c}_i$ of each $\mathbf{m}_i$ and $\mathbf{d}$, where $\mathbf{c}_{i,j} = g^{\mathbf{m}_i^j} h^{\mathbf{d}\mathbf{r}_{i,j}}$. Given $\mathbf{c}_i$ and each coefficient vectors, the cloud can compute and aggregate the result as,

$$\begin{aligned}\mathbf{v} &= \prod_{j=0}^{k_1}\left(g^{\mathbf{m}_1^j}\cdot(h^{\mathbf{d}\mathbf{r}_{1,j}})\right)^{-a_{k_j}} \cdots \prod_{j=0}^{k_z}\left(g^{\mathbf{m}_z^j}\cdot(h^{\mathbf{d}\mathbf{r}_{z,j}})\right)^{-z_{k_j}}\\ &= \prod_{j=0}^{k_1} g^{-a_{k_j}\cdot\mathbf{m}_1^j} \cdots \prod_{j=0}^{k_z} g^{-z_{k_j}\cdot\mathbf{m}_z^j} \cdot h^{-\bar{\mathbf{r}}_1\mathbf{d}-\cdots-\bar{\mathbf{r}}_z\mathbf{d}}\\ &= g^{-f(\mathbf{m_1})-\cdots-f(\mathbf{m_z})} \cdot h^{-\mathbf{d}\sum_{i=1}^{z}\bar{\mathbf{r}}_i} = g^{-F(\mathbf{m})}h^{-\mathbf{d}\bar{\mathbf{r}}}\end{aligned}$$

where $\bar{\mathbf{r}}_i = \sum_{j=0}^{k_i} \mathbf{r}'_{i,j}$ is the summation of inner product on $\mathbf{r}_{i,j}$ and the coefficients of each $f_i(x_i)$. Then, Alice constructs the computation factor $\lambda_i = \frac{1}{(x_i - \mathbf{m}_i)\cdot\mathbf{d}}$. Different from the basic scheme, the cloud will compute λ' based on the aggregated random number $\bar{\mathbf{r}} = \sum_{i=1}^{z} \bar{\mathbf{r}}_i$ as $\lambda'_i = \frac{1}{(x_i - \mathbf{m}_i)\cdot\mathbf{d}\cdot\bar{\mathbf{r}}}$, where $x_i \in_R Z_n$ is a set of random points on each $f_i(x)$. For each λ'_i and $\mathbf{v}_i$, the cloud performs the same procedure as in our basic approach to generate a set of δ_i and sends to Alice. Then, Alice can use the same way to derive the value of $F(\mathbf{m})$ from $\mathbf{v}$. Then, Alice is able to obtain the proving vector $\mathbf{W} := g^{\sum_{i=1}^{z}\sum_{j=0}^{k_i-1} w_{i,j} x_i^j}$ from the company by sending $(\vec{x}, F(\mathbf{m}))$. To verify the computation results, Alice first computes $(g^{\bar{\mathbf{r}}})^{\mathbf{d}}$ and $\eta = (h^{\bar{\mathbf{r}}})^{-\mathbf{d}\sum_{i=1}^{z}\frac{1}{x_i - \mathbf{m}_i}}$. Then, she aggregates δ_i and check the equality of the following equation,

$$e(\mathbf{W} \cdot \eta, g) \overset{?}{=} e(\prod_{i=1}^{z} \delta_i, g^{\mathbf{d}\bar{\mathbf{r}}})$$

Proof. **Correctness:** We prove the quality of two sides

$$\begin{aligned}\mathtt{LHS} &= e(g^{\sum_{i=1}^{z}\sum_{j=0}^{k_i-1} w_{i,j}x^j}, g)e(\eta, g)\\ &= e(\prod_{i=1}^{z} g^{w_i(x_i)}, g)e(h^{-\mathbf{d}\bar{\mathbf{r}}\sum_{i=1}^{z}\frac{1}{x_i-\mathbf{m}_i}}, g)\\ \mathtt{RHS} &= e(g^{\sum_{i=1}^{z}\frac{F(\vec{x})-F(\vec{\mathbf{m}})}{x_i-\mathbf{m}_i}} h^{-\mathbf{d}\bar{\mathbf{r}}\sum_{i=1}^{z}\frac{1}{x_i-\mathbf{m}_i}}, g)\end{aligned}$$

We have shown the equality of both sides based on the assumption that $F(\vec{x}) - F(\vec{\mathbf{m}}) = \sum_{i=1}^{z} w_i(x_i)(x_i - \mathbf{m}_i)$, where $w_i(x_i)$ is $(k_i - 1)$-degree polynomial. □

As we can see from the above derivation process, the total computational cost on the verification reduced from $O(z)$ to $O(1)$ on pairing operation. Meanwhile, the advanced scheme reduces the communication overhead on transmitting fewer intermediate computation results.

V. Protocol Evaluation

A. Security and Privacy Analysis

1) Correctness of Verification: First, we focus on the identity verification. The correctness of identity verification requires that patients have to show their valid certificates σ to enable the cloud to publicly verify their identities. The original blind signature verification process can only verify the provided signature Ψ generated and signed by the company (registered patients with the company), but it fails to verify whether the patient's identity has been verified by TA. Our scheme embeds the checking equation $e(g^{id_A}g^s, \sigma) = e(g, g)$ into the verification of Eq.1, so that the cloud can check the registration with the company and patients' identities at the same time.

Next, the correctness of the computation results relies on the computation on the long division of polynomials. The cloud may mistakenly compute the incorrect results $f^*(m)$ based on the input of m. First, the cloud will not be able to know the value of m from λ due to the fact that the result is the residue after module n. Second, the company computes the long division which will output $g^{w^*(x)}$, where $w^*(x)$ is the polynomial with coefficients $w_i^*, 0 \leqslant i \leqslant k-1$. However, when Alice tries to verify the correctness of computation results, the following equation

$$f(x) - f^*(m) = (x - m)w^*(x) + \varepsilon$$

has the residue $\varepsilon \neq 0$, which indicates the inequality of Eq.4.

2) Identity Privacy: According to our design objective, the company should not be able to learn the certificate σ when it generates the partially blind signature. To generate Ψ, Alice transmits $u = \alpha^{-1}H_0(\sigma||\phi') + \beta$ to the company, where σ is perfectly hidden in u by hashing and adding two random numbers α, β. Meanwhile, the company cannot link the identity with the agreed string l as either patients could use frequently changed pseudonyms to communicate with the company, or they can use same string for obtaining the same monitoring program. Another possible identity privacy leakage may happen during the verification process with the cloud. Instead of directly revealing the private certificate σ, our scheme first enables Alice to generate a set of commitments and proof, both of which are constructed with random numbers μ_i, ν_i only known to Alice. The cloud even cannot distinguish the same patient from different queries on his/her commitments due to discrete logarithm (DL) is assumed to be hard in G when message space is defined large. It also fails to reconstruct the private certificates by comparing different queries, because the commitment scheme is unconditionally hiding and computationally binding under DL assumption.

3) Delegated Program Privacy: Attackers will observe and try to obtain the detail of delegated programs $f(\vec{x})$ during the scheme run. Taking our basic scheme as an illustration, Alice first chooses a random set $\vec{r}$ and obtain $g^{\bar{r}}$, $h^{\bar{r}}$ from the company. Attackers would not be able to derive $\bar{r}$ from $g^{\bar{r}}$ and $h^{\bar{r}}$ due to the DL assumption, which indicates the infeasibility on deriving the coefficient $\vec{a}$ of the polynomial. In our advanced scheme with efficient updates, Alice will generate another set of $\vec{r}$ for the company to use. However, it is still infeasible for eavesdroppers to compare the difference between $g^{\bar{r}_i}$ and $g^{\bar{r}_{i+1}}$ and obtain the correlation between the coefficients and the corresponding random numbers due to the DL assumption.

4) Reported PHR Privacy: The cloud and eavesdroppers will eagerly look for the raw PHR data as we clarify in the adversarial model. When the cloud obtains $\bar{r}$ associated with $\vec{a}$, it cannot derive each r_i to launch attacks on m, because there is only one equation $\sum_{i=0}^{k} a_i r_i = \bar{r}$ but with k unknown factors. Without knowing the program details, it is infeasible for eavesdroppers to obtain raw data even they can obtain the computation results. The same situation happens on both our advanced schemes, in which the cloud cannot derive r_{i+1} from the summation $\bar{r}_{i+1}$ due to the increment of unknown k factors on each update (only one new equation obtained). The cloud or eavesdroppers would also not be able to derive m from $c_i := g^{m^i}h^{dr_i}$ due to the hardness of DL assumption and large space for random numbers (1024 bits in the following simulation settings). For the same reason, the cloud would not be able to obtain $f(m)$ from v. Meanwhile, since the cloud does not know the factorization of n, it cannot recover the $f(m)$ due to the subgroup decision problem is assumed hard. During the efficient update process, Alice sends $g^{m_{i+1}^i - m_i^i}$ to the cloud. However, the cloud cannot identify the updated message. For example, it cannot know $m_{i+1}+m_i$ and $m_{i+1}-m_i$ given $g^{m_{i+1}^2 - m_i^2}$ and $g^{m_{i+1}-m_i}$ due to the hardness of Computational Diffie-Hellman (CDH) assumption. So, the cloud cannot obtain either m_{i+1} or m_i via linear computation.

B. Efficiency Analysis

1) Simulation-based Analysis: First, we use Pairing-based Cryptography (0.5.12) Library to implement our simulation on computational cost. We take Tate pairing as our basic pairing operation. The elliptic curve we use for the our scheme is Type A1. A curve of such type has the form of $y^2 = x^3 + x$ supporting the composite group pairing, in which we set $|n| = 1024$ bits and $|p| = |q| = 512$ bits. The base field of the curve is 1024 bits with embedding degree of 2, and the order of group is 160 bits, which gives the same security level as 1024-bit RSA. For the simulation-based analysis, we use a laptop with an Intel processor 2.8GHz and 4GB RAM under the platform Ubuntu 11.10.

- *Privacy-preserving Identity Verification*

The privacy-preserving identity verification mainly involves patients, the company, and the cloud. For the evaluation on the computational cost, we focus on discussing the patient and the cloud side. To generate the partially blind signature Ψ, the patient has to compute 5 exponentiation (exp) operations and 2 multiplication (mul) operations on G. The commitment and proof generation process take 16-exp and 16-mul in total. On the cloud side, it takes about 5 pairing operations as $e(g, g)$ is assumed to be a publicly known factor.

As we can see from Table.I, the total cost for privacy-preserving identity verification is 950.4 ms for patients, and the verification costs the cloud 369.3 ms. If the cloud could preprocess (pp) some computation intensive parts in the verification, the cost will reduce to 93.97ms for the verification.

TABLE I
COMPUTATIONAL COST ON IDENTITY VERIFICATION

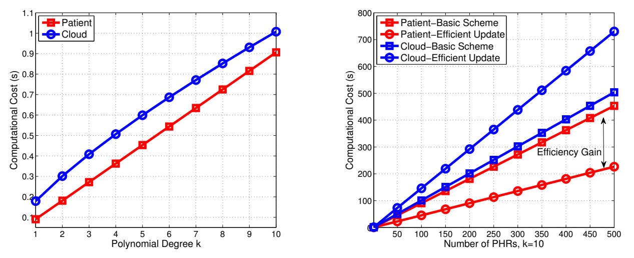
	Patient	Cloud	Cloud(pp)
Computational Cost (ms)	950.4	369.3	93.97

- *Verifiable PHR Computation*

In the verifiable PHR computation part, we mainly consider the computational cost on raw data encryption, decryption, and verification on both our basic scheme and advanced schemes. First of all, we evaluate the impact of the degree of the polynomial on the computational cost. For the simulation settings, we set the value of a_i as random numbers from a chosen uniform distribution, and the maximum of k is set to be 10 which fits for most polynomial regression. As we can see from Fig.2(a), the difference between the ciphertext generation and the PHR computation on the cloud side is not large. Especially, if we consider the cloud has more powerful computation capability, the cost on the computation could be negligible. However, for the monitoring program requiring periodic updates, the patient's cost become larger as shown in Fig.2(b). When the number of updated PHRs reach 500, the computational cost for the patient would be 453.2s. By using our advanced scheme with efficient update, we shift more than half computational load to the cloud, which results in the cost as 221.6s. Meanwhile, on the cloud side, it increases linearly as the PHRs grows, and it reaches 730.4s if we implement the efficient update on the patient side.

Then, we evaluate the computational cost on decrypting the encrypted message v. With the knowledge of the factorization of n, the patient could to decrypt v by using brute force or Pollard's lambda methods. Given specific message space of M, we have shown the computational cost in Fig.3. For the message space reaching 1000, it takes 5.66s using brute force approach and 1.43s for Pollard's lambda approach, respectively. Note for most medical programs, the range of monitored raw data value is usually smaller than 1000. By this mean, we show the efficiency of our decryption scheme on the patient's side.

We also conduct the simulation on the verification of the computational results from the cloud. First, we give our



(a) Impact of Polynomial Degree (b) Impact of Number of PHRs

Fig. 2. Computational Cost in PHR Computation

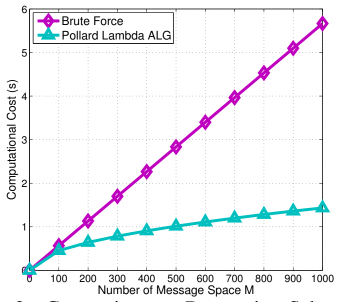
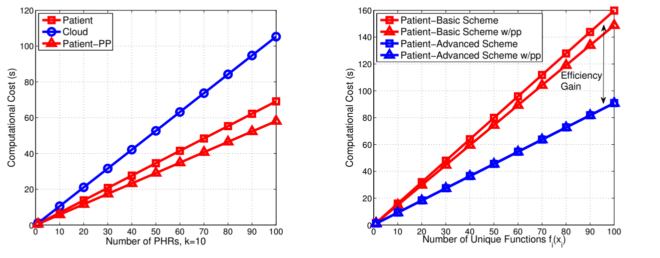


Fig. 3. Comparison on Decryption Scheme

simulation results on our basic scheme in Fig.4(a). It shows the computational costs on our basic scheme with the number of simultaneous PHRs, where the cost on the patient's side is slightly less than the cloud side due to the verification on computing the costly pairing operation. The patient consumes more than 69.1s to verify 100 PHR results for the programs with $k = 10$ and 58.1s if we implement preprocessing on the pairing operation. Then, we compare the basic scheme with our advanced scheme on multi-variable monitoring program. For the simulation, we set all $f_i(x_i)$ to have the same degree, say, 10, and $F(x)$ contains up to 100 polynomials. The basic scheme takes 159.8s to verify 100 polynomials with 100 input raw PHRs m_i, while we could apply the preprocessing on pairing operation to reduce it to 148.7s. Compared with the basic scheme, our advanced scheme allows patient to aggregate the δ_i, which shifts z multiplication operations on pairing to the multiplication operations on G. As we can see from Fig.4(b), the computational costs reduce to 90.8s and 90.6s for pairing and preprocessing pairing approaches, respectively.



(a) Total Cost for Basic Scheme (b) Total Cost for Advanced Scheme

Fig. 4. Computational Cost in Verifiable PHR Computation

2) Storage Cost Analysis: The storage analysis mainly concentrates on the patient and the cloud side. According to our simulation setting, the group G has 512 bits, and the element on Z_n is 160 bits long (the same as security parameter ξ). The partially blind signature takes $4|G| + \xi$ bits for each patient, while each one has to store the verification key g^s and the public parameters (g, g_0, h) for total $8|G| + \xi$ bits. The patients also has to store the private key $\mathtt{sk}$ for 512 bits. For each monitored PHR raw data, the patient has to store $(k+2)|G|+|\xi|$ bits for the ciphertexts and λ for the first time. Then, he/she can delete the previous ciphertexts and only keep $k|G|$ bits in each update. On the cloud side, it stores $O(Nk)$ bits for N users, and $O(zN \max\{k_z\})$ bits for multi-variable polynomial monitoring programs.

VI. Conclusion

In this paper, we propose a verifiable privacy-preserving monitoring scheme for cloud-assisted mHealth systems. The proposed scheme outsources the computation load to the cloud, and enables the cloud to not only return the computation result on encrypted monitored PHRs, but also provides the provable signature on it. Patients would be able to verify the correctness of the computation results after decrypting the ciphertext. Based on the security and efficiency analysis, we have shown the privacy preservation for patients' PHRs and delegated programs and demonstrate the feasibility of our scheme. We expect that our proposed approach can pave the way for wide deployment of mobile health monitoring technologies.

References

[1] S. M. Kelly, "In google fit vs. apple healthkit, fitness apps stay neutral." [Online]. Available: http://mashable.com/2014/06/27/healthkit-google-fit-apps/

[2] P. Mohan, D. Marin, S. Sultan, and A. Deen, "Medinet: Personalizing the self-care process for patients with diabetes and cardiovascular disease using mobile telephony," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, Aug 2008, pp. 755–758.

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

[4] Amazon, "Amazon ec2 and amazon rds service disruption." [Online]. Available: http://aws.amazon.com/message/65648/

[5] M. Li, S. Yu, K. Ren, and W. Lou, "Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings," *SECURECOMM'10*, pp. 89–106, 2010.

[6] L. Guo, C. Zhang, J. Sun, and Y. Fang, "PAAS: Privacy-preserving attribute-based authentication system for ehealth networks," in *The 32nd IEEE International Conference on Distributed Computing Systems*, ser. ICDCS 2012. Macau, China: IEEE, 2012.

[7] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient controlled encryption: ensuring privacy of electronic medical records," *Proceedings of the 2009 ACM workshop on Cloud computing security, CCSW '09*, pp. 103–114, 2009.

[8] J. Jin, G.-J. Ahn, H. Hu, M. J. Covington, and X. Zhang, "Patient-centric authorization framework for sharing electronic health records," *Proceedings of the 14th ACM symposium on Access control models and technologies*, pp. 125–134, 2009.

[9] L. Guo, C. Zhang, J. Sun, and Y. Fang, "A privacy-preserving attribute-based authentication system for mobile health networks," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 9, pp. 1927–1941, Sept 2014.

[10] J. Bethencourt, E. Shi, and D. Song, "Signatures of reputation: Towards trust without identity," *In 14th International Conference on Financial Cryptography and Data Security (FC '10)*, January 2010.

[11] J. Groth, "Fully anonymous group signatures without random oracles," *Proceedings of the Advances in Crypotology 13th international conference on Theory and application of cryptology and information security ASIACRYPT'07*, pp. 164–180, 2007.

[12] M. Belenkiy, M.Chase, M. Kohlweiss, and A. Lysyanskaya, "Non-interactive anonymous credentials," *Cryptology ePrint Archive, Report 2007/384*, 2007, http://eprint.iacr.org/.

[13] L. Guo, C. Zhang, H. Yue, and Y. Fang, "Psad: A privacy-preserving social-assisted content dissemination scheme in dtns," *Mobile Computing, IEEE Transactions on*, vol. 13, no. 12, pp. 2903–2918, Dec 2014.

[14] ——, "A privacy-preserving social-assisted mobile content dissemination scheme in dtns," in *The 32nd IEEE International Conference on Computer Communications*, ser. INFOCOM 2013. Turin, Italy: IEEE, 2013, pp. 2349–2357.

[15] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Proceedings of the 30th Annual Conference on Advances in Cryptology*, ser. CRYPTO'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 465–482.

[16] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," in *Proceedings of the 9th International Conference on Theory of Cryptography*, ser. TCC'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 422–439.

[17] R. Canetti, B. Riva, and G. N. Rothblum, "Two protocols for delegation of computation," in *Proceedings of the 6th International Conference on Information Theoretic Security*, ser. ICITS'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 37–61.

[18] D. Fiore and R. Gennaro, "Publicly verifiable delegation of large polynomials and matrix computations, with applications," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 501–512.

[19] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, ser. SP '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 238–252.

[20] C. Papamanthou, E. Shi, and R. Tamassia, "Signatures of correct computation," in *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*, ser. TCC'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 222–242.

[21] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *Advances in Cryptology-ASIACRYPT 2010*. Springer, 2010, pp. 177–194.

[22] Q. Li and G. Cao, "Providing privacy-aware incentives for mobile sensing," in *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*. IEEE, 2013, pp. 76–84.

[23] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonysense: Privacy-aware people-centric sensing," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '08. New York, NY, USA: ACM, 2008, pp. 211–224.

[24] D. Christin, C. Rosskopf, M. Hollick, L. Martucci, and S. Kanhere, "Incognisense: An anonymity-preserving reputation framework for participatory sensing applications," in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, March 2012, pp. 135–143.

[25] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *Advances in Cryptology —CRYPTO 2001*, pp. 213–229, 2001.

[26] E.-J. Goh, "Encryption Schemes from Bilinear Maps," Ph.D. dissertation, Department of Computer Science, Stanford University, Sep 2007.

[27] A. Menezes, S. Vanstone, and T. Okamoto, "Reducing elliptic curve logarithms to logarithms in a finite field," in *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing*, ser. STOC '91. New York, NY, USA: ACM, 1991, pp. 80–89.

[28] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of cryptography*. Springer, 2005, pp. 325–341.

[29] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theor.*, vol. 22, no. 6, pp. 644–654, Sep. 2006.

[30] H. Sohn, "Effects of environmental and operational variability on structural health monitoring," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 365, no. 1851, pp. 539–560, 2007.

[31] X. Chen, *Impact of Continuous Glucose Monitoring System on Model Based Glucose Control*. Master Thesis, University of Canterbury, 2007.

[32] J. De Muth, *Basic Statistics and Pharmaceutical Statistical Applications, Third Edition*, ser. Pharmacy Education Series. Taylor & Francis, 2014.

[33] R. S. B. Lee, P.R. LeDuc, "Unified regression model of binding equilibria in crowded environments," *Scientific Report*, vol. 1, no. 97, pp. 1–11, sep 2011.

[34] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," *CRYPTO'04, LNCS*, pp. 41–55, 2004.

[35] S. S. Chow, L. C. Hui, S.-M. Yiu, and K. Chow, "Two improved partially blind signature schemes from bilinear pairings," in *Information Security and Privacy*. Springer, 2005, pp. 316–328.

[36] J. Groth and A. Sahai, "Efficient non-interactive proof systems for bilinear groups," in *Proceedings of the theory and applications of cryptographic techniques 27th annual international conference on Advances in cryptology*, ser. EUROCRYPT'08, 2008, pp. 415–432.

[37] J. M. Pollard, "Monte carlo methods for index computation," *Mathematics of computation*, vol. 32, no. 143, pp. 918–924, 1978.