Volume 4, Issue 4, April 2014 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Research Two Aggregate Algorithm Techniques for Third Party Auditor (TPA) Supports to Cloud Computing

M.Balaganesh[1*],
[1] *Associate Professor,*
Sembodai Rukmani Varatharajan Engineering College, Sembodai, India

J.Venkateshan[2]
[2] *J.Venkateshan,*
P.G Scholar, Sembodai Rukmani Varatharajan Engg. College, Sembodai, India

Abstract: Healthcare information, and to some extent patient management, is progressing toward a wireless digital future.This change is driven partly by a desire to improve the current state ofmedicine using new technologies, partly by supply-and-demand economics, and partly by the utility of wireless devices. Wired technology can be cumbersome for patient monitoring and can restrict the behavior of the monitored patients, introducing bias or artifacts. However, wireless technologies, while mitigating some of these issues, have introduced new problems such as data dropout and "information overload" for the clinical team. This paper is to address this important problem and design a cloud-assisted privacy preserving mobile health monitoring system to protect the privacy of the involved parties and their data. Moreover, the outsourcing decryption technique and a newly proposed key private proxy reencryption are adapted to shift the computational complexity of the involved parties to the cloud without compromising clients' privacy and service providers' intellectual property. Finally, our security and performance analysis demonstrates the effectiveness of our proposed design. And also it describes a system known as MediNet that is being developed to personalize the healthcare process for patients with diabetes and cardiovascular disease using a cellular phone network.

Index Terms: Healthcare, key private proxy re encryption, mobile telephony, Medinet

I. INTRODUCTION

The great deployment of mobile devices, such as smart phones equipped with low cost sensors, has already shown great potential in improving the quality of healthcare services. The Microsoft launched project "MediNet" is designed to realize remote monitoring on the health status. Current telemedicine systems have created a means whereby health monitoring systems have transferred the point of care closer to the patient and have allowed the patient to be better informed and actively involved in the self-care process of diabetes and cardiovascular diseases in remote areas in Caribbean countries.

Wireless medical data transmission has also been driven by a desire to provide remote analysis of physiological data, assuming that resource allocation can be more effective in such a manner. Personalization can be defined as the modification of the nature of an activity which a user is engaged in or wants to be engaged in, to the unique characteristics of the user and the context within which the user currently resides [19]. Personalization in telemedicine is important since every patient's needs are different [1,2]. Personalized healthcare provides an opportunity to achieve more proactive treatment where problems can be addressed and hopefully prevented at the earliest possible moment. In the following sections we present an overview of the MediNet system. The components of the system are introduced and the client's interface is described in the following sections and it illustrates some personalized recommendations made to the patients. Here the reasoning engine makes use of both the group level characteristics and individual level characteristics to generate an appropriate response. Insurance Portability and Accountability Act) provide baseline protection for personal health record, they are generally considered not applicable or transferable to cloud computing environments. Besides, the current law is more focused on protection against adversarial intrusions while there is little effort on protecting clients from business collecting private information. Meanwhile, many companies have significant commercial interests in collecting clients' private health data [3] and sharing them with either insurance companies, research institutions or even the government agencies. It has also been indicated [4] that privacy law could not really exert any real protection on clients' data privacy unless there is an effective mechanism to enforce restrictions on the activities of healthcare service providers. Traditional privacy protection mechanisms by simply removing clients' personal identity information (such as names or SSN) or by using anonymization technique fails to serve as an effective way in dealing with privacy of mHealth systems due to the increasing amount and diversity of personal identifiable information [5]. It is worth noting that the collected information from an mHealth monitoring system could contain clients' personal physical data such as their heights, weights, and blood types, or even their ultimate personal identifiable information such as their fingerprints and DNA profiles [7]. According to [8], personal identifiable information (PII) is "any information, recorded or otherwise, relating to an identifiable individual. Almost any information, if linked to an identifiable individual, can become personal in nature, be it biographical, biological, genealogical, historical,

transactional, locational, relational, computational, vocational, or reputational". In other words, the scope of PII might not necessarily be restricted to SSN, name and address, which are generally considered as PII in the traditional sense. In this paper, we design a cloud-assisted mHealth monitoring system (CAM). We first identify the design problems on privacy preservation and then provide our solutions. To ease the understanding,

we start with the basic scheme so that we can identify the possible privacy breaches. We then provide an improved scheme by addressing the identified privacy problems. The resulting improved scheme allows the mHealth service provider (the company) to be offline after the setup stage and enables it to deliver its data or programs to the cloud securely. To reduce clients' decryption complexity, we incorporate the recently proposed outsourcing decryption technique [12] into the underlying multidimensional range queries system to shift clients' computational complexity to the cloud without revealing any information on either clients' query input or the decrypted decision to the cloud. To relieve the computational complexity on the company's side, which is proportional to the number of clients, we propose a further improvement, leading to our final scheme. It is based on a new variant of key private proxy reencryption scheme, in which the company only needs to accomplish encryption once at the setup phase while shifting the rest computational tasks to the cloud without compromising privacy, further reducing the computational and communication burden on clients and the cloud.

II. SYSTEM MODEL AND ADVERSARIAL MODEL

To facilitate our discussion, we first elaborate our cloudassisted mHealth monitoring system (CAM). CAM consists of four parties: the cloud server (simply the cloud), the company who provides the mHealth monitoring service (i.e., the healthcare service provider), the individual clients (simply clients), and a semi-trusted authority (TA). The company stores its encrypted monitoring data or program in the cloud server. Individual clients collect their medical data and store them in their mobile devices, which then transform the data into attribute vectors. The attribute vectors are delivered as inputs to the monitoring program in the cloud server through a mobile (or smart) device.

Fig. 1. Branching program

A semi-trusted authority is responsible for distributing private keys to the individual clients and collecting the service fee from the clients according to a certain business model such as pay-as-you-go business model. The TA can 3 be considered as a collaborator or a management agent for a company (or several companies) and thus shares certain level of mutual interest with the company. However, the company and TA could collude to obtain private health data from client input vectors. We assume a neutral cloud server, which means it neither colludes with the company nor a client to attack the other side. This is a reasonable model since it would be in the best business interest of the cloud not to be biased. We admit that it remains possible for the cloud to collude with other malicious entities in our CAM, and we leave the CAM design under these stronger models as future work. We also do not assume that an individual client colludes with other clients. Our security model does not consider the possible side-channel attack [6] due to the co-residency on share resources either because it could be mitigated with either system level protection [7] or leakage resilient cryptography [8]. CAM assumes an honest but curious model, which implies all parties should follow the prescribed actions and cannot be arbitrarily malicious. In the following, we briefly introduce the four major steps of CAM: Setup, Store, TokenGen and Query. We only illustrate the functionality of these components in this section while leaving the details in later sections. At the system initialization, TA runs the Setup phase and publishes the system parameters. Then the company first expresses the flow chart of the mHealth monitoring program as a branching program, which is encrypted under the respective directed ranching tree. Then the company delivers the resulting ciphertext and its company index to the cloud, which corresponds to the Store algorithm in the context. When a client wishes to query the cloud for a certain mHealth monitoring program, the *i*-th client and TA run the TokenGen algorithm. The client sends the company index to TA, and then inputs its private query (which is the attribute vector representing the collected health data) and TA inputs the master secret to the algorithm. The client obtains the token corresponding to its query input while TA gets no useful information on the individual query. During the last phase, the client delivers the token for its query to the cloud, which runs the Query phase. The cloud completes the major computationally intensive task for the client's decryption and returns the partially decrypted ciphertext to the client. The client then completes the remaining decryption task after receiving the partially decrypted ciphertext and obtains its decryption result, which corresponds to the decision

from the monitoring program on the clients' input. The cloud obtains no useful information on either the client's private query input or decryption result after running the Query phase. Here, we distinguish the query input privacy breach in terms of what can be inferred from the computational or communication information. CAM can prevent the cloud from deducing useful information from the client's query input or output corresponding to the received information from the client. However, the cloud might still be able to deduce side information on the client's private query input by observing the client's access pattern. This issue could be resolved by oblivious RAM technique **[16],** but this is out of the scope of this paper.

III. SOME PRELIMINARIES AND SECURITY BUILDING BLOCKS

A. Bilinear Maps

Pairing is crucial to our design, which would further serve as the building blocks of our proposed CAM. A pairing is an efficiently computable, non-degenerate function, $e : G \times G \rightarrow GT$, with the bilinearity property: $e(gr, gs) = e(g, g)rs$ for any $r, s \in Z*q$, the finite field modulo q, where G, and GT are all multiplicative groups of prime order q, generated by g and $e(g, g)$, respectively. It has been demonstrated that the proposed IBE is secure under the decisional bilinear Diffie -Hellman (DBDH) assumption (which states that in the IBE setting, given (g, ga, gb, gc, S), it is computationally difficult to decide whether $S = gabc$). Details can be found in [9].

B. Branching program

In this section, we formally describe the branching programs**,** which include binary classification or decision trees as a special case.

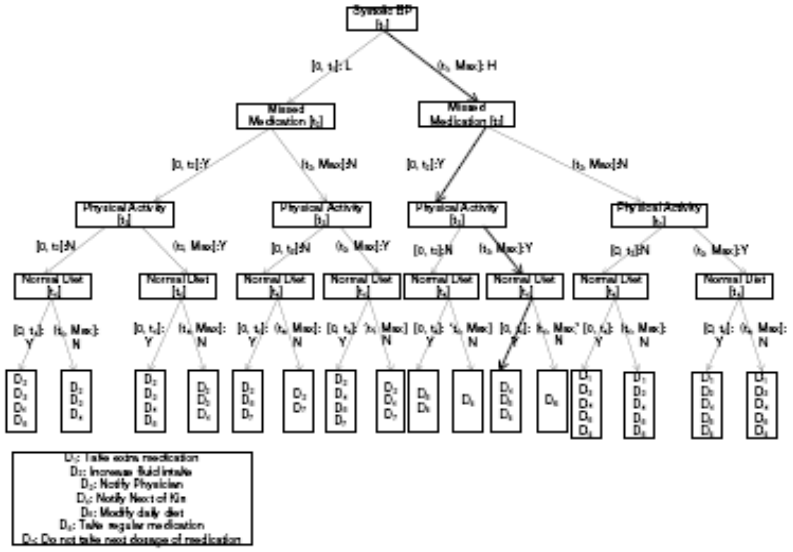

Fig. 2. Using branching program to represent a real monitoring program in MediNet project

We only consider the binary branching program (as shown in Fig. 1) for the ease of exposition since a private query protocol based on a general decision tree can be easily derived from our scheme. Let v=($v1, \cdots, vn$) be the vector of clients' attributes. To be more specific, an attribute component vi is a concatenation of an attribute index and the respective attribute value. For instance, A||KW1 might correspond to "blood pressure: 130". Those with a blood pressure lower than 130 are considered as normal, and those above this threshold are considered as high blood pressure. Each attribute value is an C-bit integer. In this paper, we choose C to be 32, which should provide enough precision in most practical scenarios. A binary branching program is a triple $\{p1, \cdots, pk\}, L,R\}$. The first element is a set of nodes in the branching tree. The non-leaf node pi is an intermediate decision node while leaf node pi is a label node. Each decision node is a pair (ai, ti), where ai is the attribute index and ti is the threshold value with which vai is compared at this node. The same value of ai may occur in many nodes, i.e., the same attribute may be evaluated more than once. For each decision node i, $L(i)$ is the index of the next node if $vai \leq ti$; $R(i)$ is the index of the next node if $vai > ti$. The label nodes are attached with classification information. To evaluate the branching program on some attribute vector v, we start with p1. If $va1 \leq t1$, set $h = L(1)$, else $h = R(1)$. Repeat the process recursively for ph, and so on, until one of the leaf nodes is reached with decision information.

C. Homomorphic encryption

Homomorphic encryption is widely used as an underlying tool for constructing secure protocols in the literature **[10],**[11]. CAM adopts a semantically secure additively homomorphic public-key encryption technique. Intuitively, for homomorphic encryption HEnc(·), given two encrypted messages HEnc($m1$) and HEnc($m2$), the encryption of the addition of the two underlying messages can be computed additively as follows: HEnc($m1+m2$) = HEnc($m1$)⋆HEnc($m2$), where ⋆ is the corresponding operation in the ciphertext space. A typical additively homomorphic encryption scheme was proposed by Paillier cryptosystem . Homomorphic encryption enables a client to obtain the token corresponding to the input attribute vectors obliviously from TA.

D. Decryption outsourcing

The pairing-based IBE system and its extensions such as attribute-based encryption has a reputation of costly decryption workload due to the bilinear paring operations in the decryption steps. Moreover, the pairing computation is considered to be especially computationally intensive for resource-constrained mobile phones. For example, for a chosen pairing function, the computation time on a PC with 2.40GHz Intel(R) Core 2 Quad, 3 GB RAM, and Windows 7 is 14.65ms while that on an Android 2.3.2 with 1GHz ARM Cortex A8 and 512 MB RAM is as high as 332.9 ms. Thus, we seek decryption outsourcing to ease the computational complexity. The decryption outsourcing in ABE was first proposed by Green *et al* [18]. It enables a client to transform his secret key to the transformation key and then delegates it to an untrusted server (e.g., a cloud) to use it to transform the original ciphertext into an El Gamal encryption of the original message. The client only needs to compute simple exponentiation operations to obtain the underlying message. In CAM, we intend to apply the outsourcing decryption technique to MDRQs based on the BF-IBE scheme. The BF-IBE based outsourcing decryption is shown as follows. AnonSetup(1_): This algorithm is exactly the same as the original BF-IBE. AnonMaskExtract(id, msk): This algorithm is performed by TA and a client. The client chooses a random number $z \in 6\ Zq$, then computes $H1(\mathrm{id})z$, and deliver $H1(\mathrm{id})z$ to TA, who will output a transformation key corresponding to id: tkid = $H1(\mathrm{id})zs$. The client keeps z as its private key skid. AnonEnc(id, *PP*,*m*): This algorithm is exactly the same as the original BF-IBE and output $C\mathrm{id} = (c1, c2, c3)$. Transform($C$id, tkid): This algorithm is performed by the cloud. The cloud parses $C\mathrm{id} = (c1, c2, c3)$ and then computes $w = e(\mathrm{tkid}, c1)$. Then it outputs the transformed ciphertext $C'\mathrm{id} = (c'1, c'2, c'3) = (w, c2, c3)$. AnonMaskDecryption($C'$id, z): This algorithm is performed by the client. Upon receiving the input of a ciphertext C'id under id together with his secret z, the client parses $C'\mathrm{id} = (c'1, c'2, c'3)$ and compute $u = c'1\ 1{=}z$, then recovers $\sigma = c'2 \oplus H2(u)$. Then the message m can be obtained by $m = c'3 \oplus H4(\sigma)$. It can be easily verified that the above scheme is indeed correct. We observe that in this construction the client only needs to compute one exponentiation in order to obtain the message, and the costly pairing operation is completed by the cloud. It can be shown as done in [25] that our proposed BFIBE with outsourcing decryption is secure against replayable chosen ciphertext attack (CCA), which implies that the following mask privacy: TA obtains no useful information on the client's identity id since $H1(\mathrm{id})z$ is just a random element to TA under random oracle model. Neither does the cloud obtain any useful information on the client's decryption result or the client identity id since the transformation key tkid = $H1(\mathrm{id})zs$ reveals nothing on id either.

E. The Notion of Key-Private PRE and Prior Constructions

In this section, we examine the half-dozen existing proxy re-encryption schemes and discuss why they do not satisfy the notion of key-privacy. Let us _rst sketch the privacy notion wanted. Intuitively, we want to capture the strong guarantee that even an active proxy colluding with a set of malicious users in the system cannot learn from the re-encryption key the identity of the involved participants nor the contents of their encrypted messages. Informally, the key-privacy game works as follows. First, the adversary is given the public keys of all honest users and the keypairs of all corrupt users. Next, the adversary is allowed to query two oracles an arbitrary number of times. The adversary may either request: (1) to have a chosen ciphertext under any user i re-encrypted to any user j or (2) to obtain the re-encryption key that translates ciphertexts from any user i to any user j. These oracles will operate regardless of the corruption status of i or j. Finally, the adversary must output a challenge pair of honest users (i_; j_), with the restriction that the adversary cannot have asked for this key before. The challenger will then return either the re-encryption key from i_ to j_ or a random key in the key space. The adversary wins if he can distinguish these cases with non-negligible probability.Before discussing the problems with speci_c PRE constructions, let's get a better sense of what cannot possibly work. In this, we point out that no deterministic re-encryption algorithm can satisfy the key-privacy de_nition. To see this, consider the generic attack where an adversary asks for a re-encryption of ciphertext C under user i to user j to obtain output C0. The adversary can then challenge on (i; j) and apply the returned re-encryption key to C. Since the re-encryption algorithm is deterministic, this should result in output C0 if this is a proper key from i to j and is unlikely to do so for a random key. Unfortunately, theist four (out of six) prior PRE schemes have deterministic re-encryption algorithms, and thus cannot be key-private. Similarly, in Section 2.1, we show that for a PRE scheme to be key-private (that is, one cannot distinguish the participants from seeing the key), the underlying encryption scheme must also be key-private in the sense of Bellare, Boldyreva, Desai and Pointcheval [3] (that is, one cannot distinguish the recipient from seeing the ciphertext). Some of the schemes also fail to have this property; mainly because they are in a bilinear setting, where the map can be used for this test. Let us now discuss some speci_cs of prior schemes.

IV. CAM DESIGN

We are ready to present our design *CAM:* cloud-assisted privacy preserving mHealth monitoring system. To illustrate the fundamental idea behind this design, we start with the basic scheme, and then demonstrate how improvements can be made step-by-step to meet our design goal. Some of the variables in the following illustration may have already been defined in the previous sections. The system time is divided into multiple time periods, called *slots*, each of which can last a week or a month depending on specific application scenarios. There is an estimated maximum number of users N requesting access to the monitoring program in any given slot. When a client attempts to access the program, it is assigned an index $i \in [1,N]$ by TA.

A. Basic CAM

The following basic scheme runs the BF-IBE system as a sub-routine and is the fundamental building block in our overall design. Setup: This algorithm is performed by TA, which publishes the system parameters for the BF-IBE scheme.

1.Store: This algorithm is performed by the company. For each node p*j* whose child nodes are not leaf nodes, the company runs *CL*(*j*) = AnonEnc(id, *PP*,*L*(*j*)) and *CR*(*j*) = AnonEnc(id, *PP*,*R*(*j*)) to encrypt the child node indices under id with either id $\in S[0;tj]$ or id $\in S[tj+1;Max]$, respectively. When the child nodes of p*j* are leaf nodes, the company generates the ciphertext as *CL*(*j*) = AnonEnc(id, *PP*,*mL*(*j*)) and *CR*(*j*) = AnonEnc(id, *PP*,*mR*(*j*)), where *mL*(*j*) and *mR*(*j*) denote the attached information at the two leaf nodes, respectively. All the generated ciphertexts are delivered and stored in the cloud.

2. TokenGen: To generate the private key for the attribute vector v=(*v*1, · · · , *vn*), a client first computes the identity representation set of each element in v and delivers all the *n* identity representation sets to TA. Then TA runs the 7 AnonExtract(id, msk) on each identity id $\in Svi$ in the identity set and delivers all the respective private keys sk*vi* to the client.

3 Query: A client delivers the private key sets obtained from the TokenGen algorithm to the cloud, which runs the AnonDecryption algorithm on the ciphertext generated in the Store algorithm. Starting from p1, the decryption result determines which ciphertext should be decrypted next. For instance, if $v1 \in [0, t1]$, then the decryption result indicates the next node index *L*(*i*). The cloud will then use sk*v*(*L*(*i*)) to decrypt the subsequent ciphertext *CL*(*i*). Continue this process iteratively until it reaches a leaf node and decrypt the respective attached information.

B. CAM with Full Privacy Preservation

The basic scheme has the following security weakness:first, the identity representation set for a client's attribute vector v is known to TA, and hence TA can easily infer all the client's private attribute vector. Second, the client cannot protect his privacy from the cloud either because the cloud can easily find out the identity representation for the private key sk*vi* , $i \in [1, n]$ by running identity test in MDRQs. The cloud can simply encrypt a random message under any attribute value v' until when it can use sk*vi* to successfully decrypt the ciphertext, which means there is a match between $v' = vi$ and hence it successfully finds out *vi*. Third, neither can the data privacy of the company be guaranteed since the identity representation of the respective range is revealed to the cloud whenever the decryption is successful due to the match revealing property (see Sec. III-D) of MDRQs. The cloud can finally figure out most of the company's branching program since it has the private keys of all the system users.

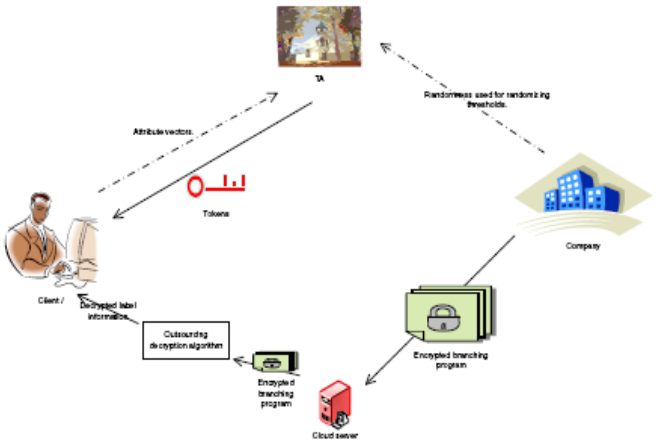

Fig. 3. CAM with full privacy of preservation

C. Final CAM with Full Privacy and High Efficiency

Since the final scheme is based on the newly proposed key private proxy re-encryption scheme, we will present this scheme first.

Key private proxy re-encryption scheme: The proxy reencryption scheme consists of the following six algorithms. Setup(1_): This algorithm is performed by TA. Upon receiving the input of the security parameter 1_, TA outputs the system parameter (G,G*T* , *q*, *g*,*Hi*, *i* = 1, 2, 3, 4, 5), the key pair for TA (*pk*, msk) = (*y*, *s*) = (*gs*, *s*), where G,G*T* are bilinear groups of prime order *q*, *g* is a random primitive root in G, *Hi*, ($i \in \{1, 2, 3, 4, 5\}$) are cryptographic hash functions. $H1 : \{0, 1\}* \rightarrow$ G, $H2 : G \times G \rightarrow Z*q$, $H3 : M \times M \rightarrow Z*q$, $H4 : GT \rightarrow M \times M$, and $H5 : G\bullet \sim M\bullet \sim M \rightarrow$ G. The system parameter is included in the following algorithms implicitly.

Ext(id, msk): This algorithm is performed by TA and a client. Upon receiving the input of an identity id, the client first picks a random number $z \in Z*q$, computes $u1 = H1(id)z$ and sends to TA. TA outputs the transformation key corresponding to id: $u2 = us$ 1 where *s* = msk and sends it back to the client. Then the client computes his private key $skid = u1 = z\ 2 = H1(id)zsz-1 = H1(id)s$. We note that TA obtains no information on the client identity because $H1(id)z$ is just a random group element under random oracle model. The transformation key can be publicly distributed due to the same reason [9] ReKey(id1, id2, msk): This algorithm is performed by TA. Upon receiving the request from delegator *D* of re-encryption from id1 to id2, it first runs the Ext algorithm on id2 to generate skid2 . Then it outputs the re-encryption key from id1 to id2: *rk*id1;id2 = (*rk*(1) id1;id2, *k*(2) id1;id2)= (*H*1(id1)*s* · *gH*2(skid2 ||*N*id1;id2),*N*id1;id2) where

*N*id1;id2 is a random element from G. Enc(id,*m*): This algorithm is performed by the company. Upon receiving the input $m \in M$, an identity id, it outputs the ciphertext $C = (c1, c2, c3)$, where $r = H3(m||\sigma)$, $c1 = gr$, $c2 = (\sigma||m) \oplus H4(e(H1(\text{id}), y)r)$, $c3 = H5(c1||c2)r$ where σ is a random element from M, the message space.
ReEnc(*C*id1 , *rk*id1;id2): This algorithm is performed by the proxy. Upon receiving the input of an original ciphertext *C*id1 = (*c*1, *c*2, *c*3) under identity id1, and a re-encryption key *rk*id1;id2 from id1 to id2, if $e(c1,H5(c1||c2)) = e(g, c3)$ holds, then it outputs the re-encrypted ciphertext *C*id2 = (*c*′1, *c*2, *c*′3, *c*4) with $c'1 = e(g, c1)$, $c'3 = e(c1, rk(1)\ \text{id1;id2})$, and $c4 = rk\text{id1;id2}$. Otherwise, it outputs $\perp$. Dec(skid,*C*id): This algorithm is performed by a client. Upon receiving the input of a ciphertext *C*id under id, and a private key skid, the algorithm is shown as follows.

1) If *C*id is an original ciphertext (*c*1, *c*2, *c*3), compute $c2 \oplus H4(e(\text{skid}, c1)) = (\sigma||m) \oplus H4(e(H1(\text{id}), y)r) \oplus H4(e(H1(\text{id})s, gr) = \sigma||m$

If $c1 = gH3(\_||m)$ and $c3 = H5(c1||c2)H3(\_||m)$ both hold, output *m*; otherwise, output $\perp$.

2) If *C* is a re-encrypted ciphertext (*c*′1,*c*2, *c*′3, *c*4) (assume that the receiver of the re-encrypted ciphertext is id′), compute $H4\ (c'3c'1H2(s\text{kid}'||c4)) \oplus c2$

$$= \left(\frac{H4\ (e(y,H1(\text{id})r) \cdot e(g, g)r \cdot H2(s\text{kid}'||N\text{id;id}')}{(e(g, g)r)H2(s\text{kid}'||N\text{id;id}'))} \right)$$

$$\oplus(\sigma||m) \oplus H4(e(H1(\text{id}), y)r) = \sigma||m$$

If $c'1 = e(g, g)H3(\_||m)$ holds, output *m*; otherwise, output $\perp$.

The last step can be omitted if only chosen ciphertext attack (CPA) security is considered. The CPA security[2] is sufficient in practice assuming there is a secure and authenticated channel between the company and the cloud.

V. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

A. Security

In our CAM, we observe that the cloud obtains no information on either the individual query vector v or the company diagnostic branching program as in our first improvement. The cloud obtains no information on the company's branching program due to the semantic security of the proxy reencryption and symmetric key encryption scheme.

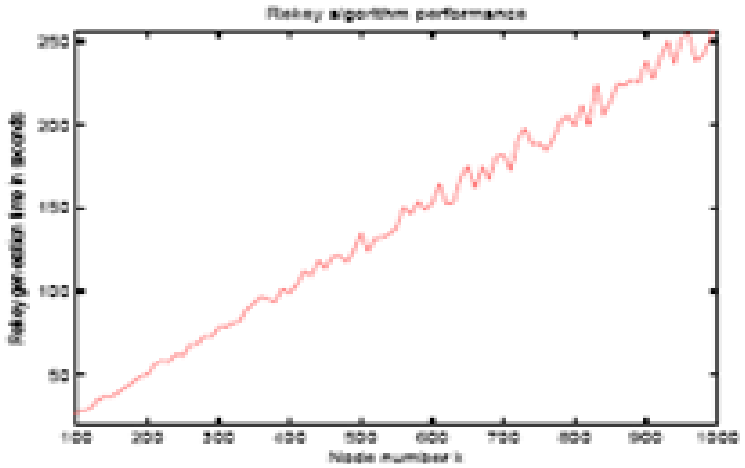

Fig:4. The security performance

The key privacy can guarantee that the cloud obtains no useful information on the branching program while completing all the computationally intensive encryption operations for the company. As in the first improvement, the transformation key contains no information on a client's query vector due to the mask privacy, which defeats the cloud's attack through performing the identity testing. We have also carried out formal analysis in the appendix to show that our proposed key private re-encryption scheme is secure and privacy-preserving under random oracle model and under decisional bilinear Diffie-Hellman (DBDH) assumption, and demonstrate that our CAM can indeed achieve our design goal.

B. Efficiency

All the timing reported below are averaged over 100 randomized runs. We assume a maximum of $k = 1000$ nodes in the branching program, which can express most complicated decision support systems as used in the MediNet [11] with 31 nodes (Fig. 2). The attribute vector has a maximum of $n = 50$ attributes, which contain much richer information than the System user n Numberr

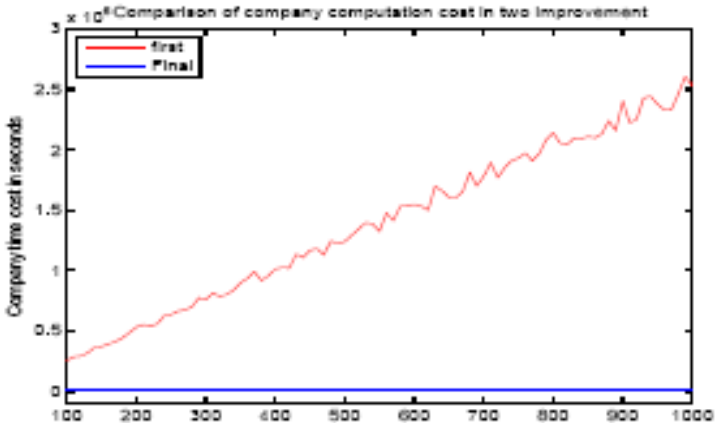

Fig. 5. Comparisons of computations

VI. Related Works

All the existing solutions require the client run multiple instances of oblivious transfer protocol with the company after setup phase, which means the company has to stay online constantly. All the current solutions [15] [16], [18] are based on garbled circuits, which implies a client must download the whole circuit to his device and complete the decryption on his own. Besides, the private computation or processing of medical information over the cloud has also attracted attention from both the security community [16] and signal processing community [17], [15]. These works can be divided into two categories: providing a solution for a specific scenario such as private genomic test [16] or private classification of users' electrocardiogram (ECG) data [17], or proposing a general framework for private processing of monitored data [17] or electronic health records [18]. Although these schemes are based on cloud computing, they do not emphasize on how to transfer the workload of the involved parties to the cloud without violating the privacy of the involved parties. Since our application scenario assumes the clients hold relatively resource-constrained mobile devices in a cloud assisted environment, it would be helpful if a client could shift the computational workload to the cloud. However, there seems no trivial approach to outsourcing the decryption of garbled circuit currently. Our proposed system adopts the recently proposed decryption outsourcing to significantly reduce the workload of both the company and clients by outsourcing the majority of the computational tasks to the cloud while keeping the company offline after the initialization phase.

VII. CONCLUSION

In this paper, we design a cloud-assisted privacy preserving mobile health monitoring system, called CAM, which can effectively protect the privacy of clients and the intellectual proerty of mHealth service providers. To protect the clients' privacy, we apply the anonymous Boneh-Franklin identitybased encryption (IBE) in medical diagnostic branching programs. To reduce the decryption complexity due to the use of IBE, we apply recently proposed decryption outsourcing with privacy protection to shift clients' pairing computation to the cloud server. To protect mHeath service providers' programs, we expand the branching program tree by using the random permutation and randomize the decision thresholds used at the decision branching nodes. Finally, to enable resourceconstrained small companies to participate in mHealth business, our CAM design helps them to shift the computational burden to the cloud by applying newly developed key private proxy re-encryption technique. Our CAM has been shown to achieve the design objective.

REFERENCES

[1] M. Balaganesh, R. Sureshkumar and J. Venkateshan, "A Survey on Techniques for Third Party Auditor in Cloud Computing", International Journal of Emerging Technology & Advanced Engineering (ISSN 2250-2459, ISO 9001:2008 Certified Journal), Volume 3, Issue 12, December 2013.

[2] A. Tsanas, M. Little, P. McSharry, and L. Ramig, "Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 4, pp. 884–893,Apr. 2010.

[3] M. Barni, P. Failla, R. Lazzeretti, A. Sadeghi, and T. Schneider, "Privacy-preserving ECG classification with branching programs and neural networks," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 2, pp. 452–468, Jun. 2011.

[4] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Proc. CRYPTO*, 2006, pp. 290–307.

[5] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.

[6] A.C. Yao. Protocols for secure computations. In FOCS, pages 160{164, 1982.

[7] M. Layouni, K. Verslype, M. Sandikkaya, B. De Decker, and H. Vangheluwe, "Privacy- preserving telemonitoring for ehealth," *Data and Applications Security XXIII*, pp. 95–110, 2009.

[8] G. Danezis and B. Livshits, "Towards ensuring client-side computational integrity," in *Proc. 3rd ACM Wo*

[9] W. Hu. Lattice scheduling and covert channels. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P'92)*, pages 52–61, 1992.
[10] C. H. Rowland. Covert channels in the TCP/IP protocol suite. *First Monday*, 2, 1997.
[11] B. W. Lampson. A note on the confinement problem.*Communications of the ACM*, 16:613– 615,1973.
[12] F. G. G. Meade. A guide to understanding covert channel analysis of trusted systems. Manual NCSC-TG-030, U.S. National Computer SecurityCenter, 1993.
[13] T. Kim, M. Peinado, and G. Mainar-Ruiz, "Stealthmem: System-level protection against cache- based side channel attacks in the cloud," in *Proc. 21st USENIX Conf. Security Symp.*, 2012, pp. 11–11, USENIX Association.
[14] E. Shi, J. Bethencourt, H. T.-H. Chan, D. X. Song, and A. Perrig, "Multidimensional range query over encrypted data," in *Proc. IEEE Symp. Security and Privacy*, 2007, pp. 350–364.
[15] I. Blake and V. Kolesnikov, "Strong conditional oblivious transfer and computing on intervals," *Advances in Cryptology-ASIACRYPT 2004*, pp. 122–135, 2004.
[16] J. Domingo-Ferrer, "A three-dimensional conceptual framework for database privacy," *Secure Data Manage.*, pp. 193–202, 2007.
[17] T. Lim, *Nanosensors: Theory and Applications in Industry, H*
[18] P. Ohm, "Broken promises of privacy: Responding to the surprising failure of anonymization," *UCLA Law Rev.*, vol. 57, p. 1701, 2010.
[19] 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 *Proc. 30th Ann. Int. Conf. IEEE Engineering in Medicine and Biology Society, 2008 (EMBS 2008)*, 2008, pp. 755–758.

