PRECISE:PRivacy-prEserving Cloud-assisted quality Improvement Service in healthcare

Feng Chen, Samuel Cheng
School of Electrical and Computer Engineering
University of Oklahoma, Tulsa, OK, 74135
USA
Email:{achenfengb, samuel.cheng}@ou.edu

Noman Mohammed
School of Computer Science
McGill University, Montreal,
Montreal, Quebec, H3A 0G4
Canada
Email:
noman.mohammed@mail.mcgill.ca

Shuang Wang，Xiaoqian Jiang
Division of Biomedical Informatics
University of California, San Diego,
La Jolla, CA 92093
USA
Email:{shw070, x1jiang}@ucsd.edu

***Abstract*—Quality improvement (QI) requires systematic and continuous efforts to enhance healthcare services. A healthcare provider might wish to compare local statistics with those from other institutions in order to identify problems and develop intervention to improve the quality of care. However, the sharing of institution information may be deterred by institutional privacy as publicizing such statistics could lead to embarrassment and even financial damage. In this article, we propose a Privacy-preserving Cloud-assisted quality Improvement Service in healthcare (PRECISE), which aims at enabling cross-institution comparison of healthcare statistics while protecting privacy. The proposed framework relies on a set of state-of-the-art cryptographic protocols including homomorphic encryption and Yao's garbled circuit schemes. By securely pooling data from different institutions, PRECISE can rank the encrypted statistics to facilitate QI among participating institutes. We conducted experiments using MIMIC II database and demonstrated the feasibility of the proposed PRECISE framework.**

Keywords—quality improvement;homomorphic encryption; garbled circuit;data privacy; cloud computing.

I. INTRODUCTION

Hospital quality is important to the reputation and financial sustainability of a hospital. Metrics such as infection rate and readmission rate reflect the quality of care. In order to improve the quality, it is necessary to compare local statistics with those from other hospitals to know what intervention needs to be prioritized. However, sharing of such statistics can be embarrassing and financially disadvantageous, which deters hospitals to exchange "sensitive" statistics. It would be beneficial if a mechanism allows hospital administrators compare these statistics to obtain a ranking without disclosing the underlying statistics.

This is closely related to the famous Yao's millionaire problem, where two millionaires want to know who is richer without disclosing their total asset to the other[1], [2]. The problem for comparing measurements related to hospital quality is challenging as we need to consider a multi-institution comparison scenario to protect intermediary information exchange.

A. Motivating Example

Imagine several hospitals, which are located at different locations, want to study morbidity related to the bloodstream infection (BSI) in Emergency Room (ER) and improve the quality of care. They would like to know the morbidity ranking in terms of BSI morbidity stratified by age, gender, staff training, vascular access care audits, etc. Such ranking can help hospitals gain insights to identify necessary intervention for improvement but it should not disclose sensitive information from individual hospitals. For example, Hospital A might find that the less frequent vascular access care leads to higher ranked BSI morbidity in the elder population, for which intervention can be developed to improve the quality. Our framework can support such comparison in a privacy-preserving manner. Before we elaborate the details, let us review related methodology.

B. Related Techniques

Secure Multiparty Computation (SMC)[3]is one of the cryptographic techniques for securely aggregating information among different parties. However, SMC is not always practical as it requires inter-party (peer to peer) communication.

Alternatively, data perturbation based methods[4]–[8] have been proposed, which try to generalize or add noise to the raw data in order to hide the sensitive information. Among existing strategies, differential privacy based methods[4], [6], [9]have received a lot of attention as the privacy definition provides the strongest privacy protection (without making any assumption on attackers' background knowledge).However, the main drawback of perturbation based methods is that added noise may destroy the utility of the outcome (i.e., in our case, the

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

ranking orders).Order-preserving encryption [10]provides yet another workaround to conduct ranking operation on encrypted data, but it cannot support secure aggregation among distributed datasets, which is necessary for comparing local statistics with the global ones.

To address the limitations of existing techniques, we propose a privacy-preserving cloud-assisted framework for securely aggregating information from distributed data sources as well as performing global ranking in cipher text.

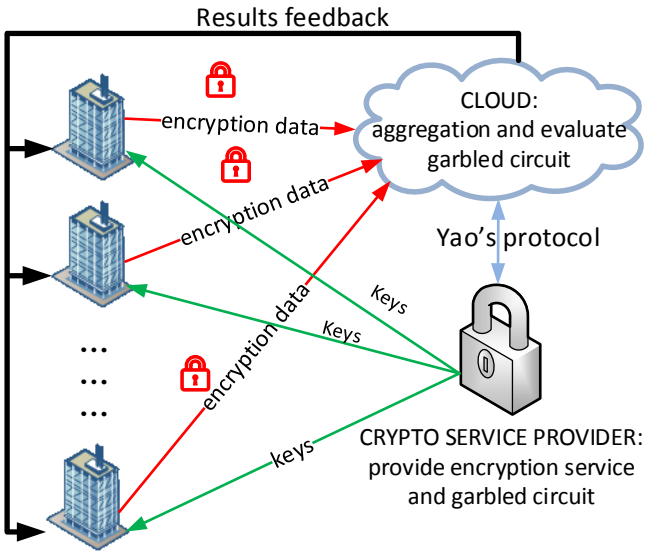


Fig. 1System overview of the proposed PRECISE framework, which includes M hospitals, a cloud service and a Crypto Service Provider (CSP).In the proposed framework, neither the cloud nor the CSP can eavesdrop sensitive information from any of the M hospitals.

II. METHODOLOGY

A. System Framework

Fig. 1illustratesthe frame work of the proposed method, which includes M hospitals, a cloud service, and a crypto service provider(CSP). The cloud service analyzes encrypted data from M different hospitals in a privacy-preserving manner, and answers aggregate queries. The CSP manages the public key (data encryption) and private key (data decryption), and it is the only entity capable of decrypting a cipher text. All parties in this study are assumed to be semi-honest, which means they follow the protocol honestly but may try to deduce additional information from the received messages during the protocol execution. To minimize privacy risk, the cloud service is restricted to only answer the rank information (e.g., which hospital has the largest morbidity for a given cohort)rather than providing the actual counts.

The workflow of the proposed frame work is summarized as follows: first, the cloud gathers encrypted counts of different query criteria from each hospital using the homomorphic encryption algorithm and adds random masks to the encrypted count of each query so that the real counts cannot be revealed by the CSP. Next, a garbled circuit is designed based on the Yao's protocol[1], by which the cloud (together with the CSP)can answer the ranking information using encrypted data.

There maining of this section is organized as follows: we briefly present the relevant cryptographic tools used in our proposed protocol. Then, we will elaborate on the implementation details of the proposed protocol.

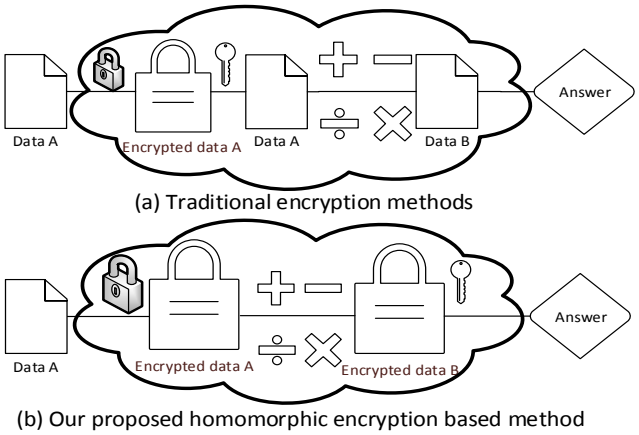


Fig. 2 Comparison between traditional encryption and homomorphic encryption methods.

B. Homomorphic Encryption

Homomorphic encryption [11], [12] is a form of encryption where a specific algebraic operation performed on the plaintext is equivalent to another algebraic operation performed on the ciphertext, and when decrypted, matches the results of the same operation performed on the plaintext. Fig. 2illustrates the difference between homomorphic encryption and traditional encryption methods. In specific, there are three types of homomorphic encryption techniques[13]: (1) partially homomorphic encryption that is generally specialized in a single type of operations (e.g., either addition or multiplication) [14]–[16], (2) leveled homomorphic encryption that operates on both operations for a limited number of iterations and an increased computational complexity[17]. (3) fully homomorphic encryption that operations on both operations without limiting the number of iteration but it also results in the highest complexity[18]–[21].For a given task, it is important to select a proper homomorphic encryption scheme to strike the right tradeoff between arithmetical flexibility and computational complexity. The addition operation is the basic primitive of our frame work. We resort to the Paillier's scheme[16], which is a partially homomorphic encryption techniques with homomorphic addition property, to conduct this operation. Let us denote by$E(x_1)$ and $E(x_2)$theencrypted cipher texts of two plaintexts x_1 and x_2 . In Paillier's homomorphic scheme, the product of two cipher text s results in the encrypted version of the summation of both plaintexts(i.e., $E(x_1) \cdot E(x_2) \bmod n^2 = E(x_1 + x_2) \bmod n$, where 'mod' denotes the modular operation and $n = pq$ is the product of two large prime numbers p and q.)

C. 1-2 oblivious transfer (OT) protocol

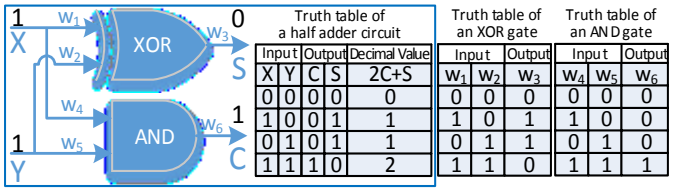
Oblivious transfer(short for 1-2 OT)protocol is a constant round communication protocol, which guarantees that Party A can obtain one of the two messages from Party B without letting Party B knows which message is actually selected. We will not go through the details of the OT protocol in this paper, But, readers can find more implementation details in[22]. The

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

OT protocol is one of the steps necessary to implement the following Yao's protocol.

D. Yao's Protocol[1]

The original Yao's protocol supposes two parties, for example, Alice and Bob, plan to compute a function $f(x, y)$, where x is owned by Alice, and y is owned by Bob. However, none of these parties would like to expose its input to the other party in evaluating the function $f(x, y)$. To satisfy this requirement, Alice will first convert the function $f(x, y)$ into a Boolean circuit, in which sever allogic gates will be specifically combined together to realize the given function. Here, each logic gate can perform a logical operation on one or more binary inputs and produces a single binary output. Fig. 3depicts an example of a half adder circuit to implement the function $f_{ha}(x, y) = x + y$ with $x, y \in \{0, 1\}$, wherethe half adder circuit includes an XOR (i.e., exclusive OR) gate and an AND gate with two inputs and two outputs. Moreover, each logic gate has three wires corresponding to two binary inputs and one binary output, where each wire will be assigned a uniqueindex w_i with $i = \{1, 2, \ldots, W\}$ and W is the total number of wires in the Boolean circuit (e.g., $W = 6$ inFig. 3).



Truth table of a half adder circuit

Input		Output		Decimal Value
X	Y	C	S	2C+S
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	0	2

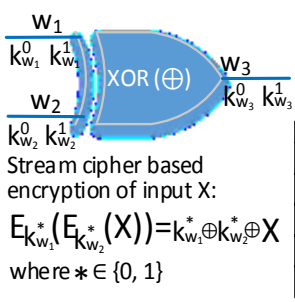
Truth table of an XOR gate

Input		Output
w_1	w_2	w_3
0	0	0
1	0	1
0	1	1
1	1	0

Truth table of an AND gate

Input		Output
w_4	w_5	w_6
0	0	0
1	0	0
0	1	0
1	1	1

Fig. 3 An example of a half adder circuit in implementing function $f_{ha}(x, y) = x + y$ with $x, y \in \{0, 1\}$, which includes an XOR gate and an AND gate. The half adder has two single binary inputs x and y and two outputs, i.e., sum (S) and carry (C), where the decimal output can be represented as $f_{ha}(x, y) = 2C + S$. Each logic gate (e.g., XOR or AND gate) has three wires, which correspond to two binary inputs (e.g., W_1 and W_2 in the XOR gate) and one binary output (e.g., W_3 in the XOR gate). The truth tables of the half adder circuit, XOR and AND gates have been shown as references.



Secret keys and check code

$k^0_{w_1}$	$k^1_{w_1}$	$k^0_{w_2}$	$k^1_{w_2}$	$k^0_{w_3}$	$k^1_{w_3}$	Check Code (CC)
1010	0000	1111	1001	1000	0101	0000

Encrypted truth table of an XOR gate

Input		Encrypted Check Code (ECC)	Encrypted Output (EO)
w_1	w_2	$k^*_{w_1} \oplus k^*_{w_2} \oplus CC = ECC$	$k^*_{w_1} \oplus k^*_{w_2} \oplus k^*_{w_3} = EO$
$k^0_{w_1}$	$k^0_{w_2}$	1010⊕1111⊕0000=0101	1010⊕1111⊕1000=1101
$k^1_{w_1}$	$k^0_{w_2}$	0000⊕1111⊕0000=1111	0000⊕1111⊕0101=1010
$k^0_{w_1}$	$k^1_{w_2}$	1010⊕1001⊕0000=0011	1010⊕1001⊕0101=0110
$k^1_{w_1}$	$k^1_{w_2}$	0000⊕1001⊕0000=1001	0000⊕1001⊕1000=0001

Fig. 4. An example of an encrypted XOR gate, where w_1 and w_2 are the input wires andw_3 is the output wire. For each wire $w_i i = \{1, 2, 3\}$, two secret keys $k^0_{w_i}$ and $k^1_{w_i}$ will be selected to represent the 0 and 1, respectively. Then, a stream cipher based encryption scheme[23] is used to generate an encrypted truth table of the XOR gate with encrypted output and check code. Here, the check code can be utilized to validate the garbled output, which will be explained later.

Next, Alice will randomly generate two secret keys (a.k.a., garbled values) $k^0_{w_i}$ and $k^1_{w_i}$ to represent 0 and 1, respectively, for each wire w_i with $i = \{1, 2, \ldots, W\}$. For example, Alice needs to generate 6 keys for the XOR gate, where the input keys can be used to encrypt the output keys as shown in the encrypted truth table in Fig. 4. A stream cipher based scheme is applied to encrypt the output keys as well as a check code. The check code can be used to validate the output keys, which will be explained later. Besides, other encryption methods (e.g., AES [24]) can also be employed to enhance the security in practice. Then, Alice randomly permutes the encrypted truth table and sends the garbled table as well as the check code to Bob as shown in Fig. 5 (a). Besides, Alice will also send the output keys to Bob.

Let's suppose the actual inputs from Alice and Bob are 0 and 1, respectively. As shown in Fig. 5 (b), Bob will obtain his garbled input $k^1_{w_2}$ from Alice using the OT protocol [22], by which Alice cannot learn what is the input from Bob. Furthermore, Alice will also send her garbled input (i.e., $GI(0) = k^0_{w_1} = 1010$ in Fig. 5 (b)) to Bob, where Bob only learns $k^*_{w_1} = 1010$ but cannot figure out if the '*' corresponds to 0 or 1. Once Bob obtains both garbled inputs $k^*_{w_1}$ and $k^1_{w_2}$, he can decrypt the garbled truth table and match the check code provided by Alice to obtain a uniquely valid output of the gate, where the output is still a garbled value. Therefore, Bob cannot find out what the true value of the output is. As gates in a circuit are connected through wires, Bob can continue evaluating all gates in the circuit one by one, where the garbled output from previous gate can be used as the garbled input for the current gate. In case of reaching the end of the circuit, Bob can decrypt the output based on the secret keys obtained from Alice. Fig. 5 (c)shows an example of evaluating a circuit with a single XOR gate using Yao's protocol.

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

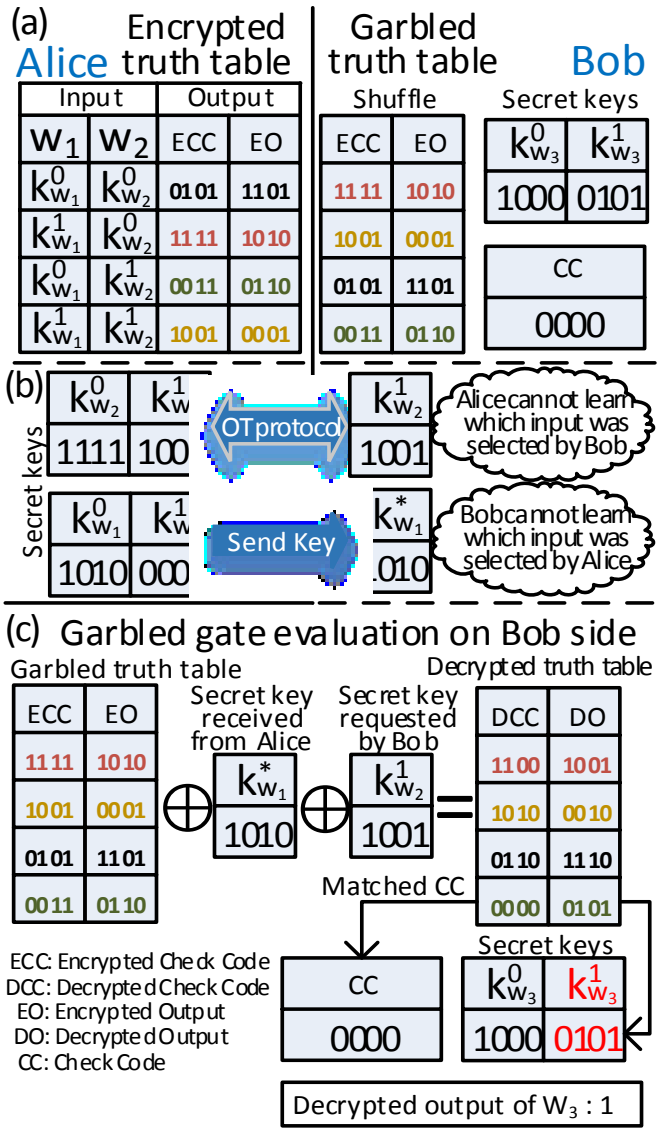


Fig. 5 An example of Yao's protocol using an XOR gate.This example is provided for illustrative purposes only, wherethe security protection cannot be achieved with this circuit. But, it can serve as the basic building blockof a more complicated circuit.See Remark I for more discussions.

Remark I: When evaluating the circuit, Bob needs to keep the encrypted intermediate outputs of each gate as secrets, so that Alice cannot infer Bob's true inputs. Moreover, the security of a circuit is based on the assumption that neither Alicen or Bob cannot derive the other's input based on the final output of a function and his/her own input .It is clear that many functions may be unable to satisfy the above assumption. For example, given the output of a single XOR gate and one input as depicted in Fig. 5 (c), one can easily infer the other input. It is worth mentioning that the example in Fig. 5 is provided for illustrative purposes only, which can serve as a basic building block for building a large garbled circuit. But users cannot achieve security protection with a single garbled XOR gate as shown in Fig. 5.In this study, we will focus on the design of a function for comparing the ranks of numerical inputs, where one cannot easily infer the original input based on the output.

A. Method Details

The general procedures of the proposed algorithm are summarized in Algorithm 1.

Lines 1-3:Each hospital H_i with $i = 1,2,\dots,M$, where M is total number of hospitals, will evaluate the same local function $f(D_{H_i})$ on its local private data D_{H_i}.Since none of the hospitals would like to expose their local sensitive results to others, each hospital H_i first encrypt its local output $f(D_{H_i})$ as $Enc_{HME}(f(D_{H_i}))$ by usinga public homomorphic encryption key k_{pub}obtained from the CSP. Then, these encrypted inputs will be sent to the cloud for global function evaluation.

Lines 4-5:The cloud will evaluate the global function $g_{fun}\left(Enc_{HME}\left(f(D_{H_1})\right), Enc_{HME}\left(f(D_{H_2})\right), \dots, Enc_{HME}\left(f(D_{H_M})\right)\right)$ over encrypted inputs. In the proposed framework, the basic cryptographic primitives that allow certain operations to be evaluated over encrypted data, include Paillier's homomorphic encryption [16], 1-2 OT [22] and Yao's garbled circuit[1].

In this step, the Paillier's homomorphic encryption scheme is used to achieve secure addition and multiplication operations over encrypted data.

1) Secure addition among homomorphic encrypted results using Paillier's homomorphic addition property.

$$g_{sum}\left(Enc_{HME}\left(f(D_{H_1})\right), Enc_{HME}\left(f(D_{H_2})\right), \dots, Enc_{HME}\left(f(D_{H_M})\right)\right) = Enc_{HME}\left(\sum_{i=1}^{M} f(D_{H_i})\right) \quad (1)$$

2) Secure multiplication between a constant and encrypted value using Paillier's homomorphic multiplication property.

$$g_{mul}\left(Enc_{HME}\left(f(D_{H_i})\right), C\right) = Enc_{HME}\left(Cf(D_{H_i})\right) \quad (2)$$

Lines 6-9: The cloud adds random masks on encrypted values based on the homomorphic addition property as follows

$$Enc_{HME}\left(f(D_{H_i})\right) + Enc_{HME}(\mu_{H_i}^1) = Enc_{HME}(f(D_{H_i}) + \mu_{H_i}^1), \quad (3)$$

$$Enc_{HME}\left(Cf(D_{H_i})\right) + Enc_{HME}(\mu_{H_i}^2) = Enc_{HME}(Cf(D_{H_i}) + \mu_{H_i}^2), \quad (4)$$

$$Enc_{HME}\left(\sum_{i=1}^{M} f(D_{H_i})\right) + Enc_{HME}(\mu_{sum}) = Enc_{HME}\left(\sum_{i=1}^{M} f(D_{H_i}) + \mu_{sum}\right), \quad (5)$$

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

Where C is a constant, $\mu_{H_i}^1$, $\mu_{H_i}^2$ and μ_{sum} are random masks generated by the cloud, H_i is the index of each hospital with $i = 1, 2, \dots, M$ for total M hospitals. Then, the cloud sends the encrypted masked data to the CSP, where these masked values will be decrypted using the CSP's private key. The CSP converts the decrypted masked values into garbled values as illustrated in the Yao's protocol, which can be denoted as $GV\left(f\left(D_{H_i}\right) + \mu_{H_i}^1\right)$, $GV\left(Cf\left(D_{H_i}\right) + \mu_{H_i}^2\right)$, and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right) + \mu_{sum}\right)$. The garbled values will be used as inputs to evaluate the garbled circuit in the next few steps. Moreover, the cloud needs to request the garbled value of each random mask from CSP through OT protocol, such as $GV(\mu_{H_i}^1)$, $GV(\mu_{H_i}^2)$ and $GV(\mu_{sum})$. The OT protocol ensures that the CSP cannot learn the underlying random mask generated by the cloud. Thus, the CSP cannot infer the masked values even after the decryption in line 8.

Lines 10-13:The cloud subtracts the garbled random mask values from the corresponding garbled masked values within the garbled circuit, by which the cloud can recover the garbled values of the original values as $GV\left(f\left(D_{H_i}\right)\right) = GV\left(f\left(D_{H_i}\right) + \mu_{H_i}^1\right) - GV\left(\mu_{H_i}^1\right)$, $GV\left(Cf\left(D_{H_i}\right)\right) = GV\left(Cf\left(D_{H_i}\right) + \mu_{H_i}^2\right) - GV\left(\mu_{H_i}^2\right)$, and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right) = GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right) + \mu_{sum}\right) - GV(\mu_{sum})$. The use of garbled circuit protect the original value from being disclosed to the cloud during the above evaluation. Then, the cloud continues evaluating a ranking garbled circuit with inputs $GV\left(f\left(D_{H_i}\right)\right), i = 1,2, \dots, M$, where the outputs are the ranking information of the underlying value $f\left(D_{H_i}\right)$ .In addition, the cloud will also evaluate a comparison garbled circuit with pair wise inputs $GV\left(Mf\left(D_{H_i}\right)\right)$ and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$, where the output is $GV\left(Mf\left(D_{H_i}\right) > \sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$. This comparison is equivalent to assess whether an input from a hospital is larger than the mean of these from all hospitals.

Lines 14-15: The cloud will match the garbled output with garbled truth table provided by the CSP to identify a valid output (see Fig. 5 (c) as an example). Finally, the decrypted results (e.g., ranking information) will be sent back to each hospital.

Remark II: The cloud should keep all the intermediate garbled values as secrets, so that the CSP cannot infer any input.

In summary, both the cloud and the CSP will be involved in the secure computation on, but none of them knows these underlying values under the proposed framework. In the next section, we will evaluate the proposed framework based on a real clinical dataset.

III. EXPERIMENT

In this section, we evaluate the proposed framework with MIMIC II Clinical Dataset[25]. The goal of the proposed framework is to perform secure data aggregation and data comparison among different hospitals in a cloud assisted environment, by which each hospital is able to retrieve its relative ranking under certain criteria in a privacy-preserving manner.

Algorithm 1 : Proposed procedures

1: **Local private data evaluation at *M* different Hospitals**
2: Each hospital H_i evaluates the same local function $f(D_{H_i})$ over its local private data D_{H_i}, where $i = 1, 2, \dots, M$
3: Each hospital H_i encrypts the result as $Enc_{HME}(f(D_{H_i}))$ based on Paillier's homomorphic encryption [16]. Then, the encrypted results will be sent to cloud service provider as inputs for further computation.
4: **Secure data aggregation:**
5: The cloud securely aggregates the inputs from each hospital based on the homomorphic addition property as depicted in (1) with the output $Enc_{HME}\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$. Then, the cloud can perform secure multiplication between a constant and the encrypted aggregation output as shown in (2)
6: **Secure conversion between homomorphic encrypted data and Garbled data**
7: The cloud adds random masks on each homomorphic encrypted value based on (3), (4) and (5), such as $Enc_{HME}\left(f\left(D_{H_i}\right) + \mu_{H_i}^1\right)$, $Enc_{HME}\left(Cf\left(D_{H_i}\right) + \mu_{H_i}^2\right)$, and $Enc_{HME}\left(\sum_{i=1}^{M} f\left(D_{H_i}\right) + \mu_{sum}\right)$. Then, the cloud sends the masked data to the CSP.
8: The CSP decrypts these masked values with its private key and converts the masked values into *garbled masked values* as illustrated in the Yao's protocol, such as $GV\left(f\left(D_{H_i}\right) + \mu_{H_i}^1\right)$, $GV\left(Cf\left(D_{H_i}\right) + \mu_{H_i}^2\right)$, and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right) + \mu_{sum}\right)$.
9: The cloud requests the *garbled random mask* values from CSP through OT protocol, such as $GV\left(\mu_{H_i}^1\right)$, $GV\left(\mu_{H_i}^2\right)$, and $GV(\mu_{sum})$.
10: **Garbled circuit evaluation:**
11: The cloud subtracts the *garbled random mask values*from the corresponding *garbled masked values* within the garbled circuit, by which the cloud can recover the garbled values of the original values as $GV\left(f\left(D_{H_i}\right)\right)$, $GV\left(Cf\left(D_{H_i}\right)\right)$ and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$.
12: The cloud continues evaluating a *ranking* garbled circuit with inputs $GV\left(f\left(D_{H_i}\right)\right)$, $i = 1,2, \dots, M$, where the outputs are the ranking information of the underlying value $f\left(D_{H_i}\right)$.
13: The cloud will also evaluate a *comparison* garbled circuit with pair wise inputs $GV\left(Mf\left(D_{H_i}\right)\right)$ and $GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$, where the output is whether $GV\left(Mf\left(D_{H_i}\right)\right) > GV\left(\sum_{i=1}^{M} f\left(D_{H_i}\right)\right)$. This comparison is equivalent to assess whether an input from a hospital is larger than the mean of these from all hospitals.
14: **Results feedback**
15: The cloud will match the garbled output with garbled truth tableprovided by the CSP to identify a valid output (see Fig. 5 (c) as an example). Finally, the decrypted results (e.g., ranking information) will be sent back toeach hospital.

We have extracted a dataset with 924 death records in intensive care unit (ICU) from the MIMIC II Clinical Database, where all the attributes used in the dataset have been listed in TABLE. The dataset includes6 categorical attributes and 1 numerical attributes.

TABLE I: MIMIC II CLINICAL DATASET

Attribute	Data Type

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

Sex	1: Female; 2: Male
Age of Death	Range from 21 to 101
Marital Status	1: Divorced; 2: Married; 3:Separated; 4: Single; 5: Unknown; 6:Widowed
Ethnicity	1:Asian; 2:Black/African American; 3:Hispanic or Latino; 4:Hispanci/ Latino-Puerto Rican; 5:Multi Race Ethnicity; 6:Other; 7:Partient Declined to Answer; 8:Unable to obtain; 9:Unknown; 10:White; 11:White-Brazilian;
Overall Payer Group	1:Auto Liability; 2:Free Care; 3: Medicaid; 4:Medicare; 5:Medicare private; 6:Other; 7:Private; 8:Self-pay;
Religion	1:7th Day Adventist; 2:Baptist; 3:Buddhist; 4:Catholtc; 5:Christian Scientist; 6:Episcopalian; 7:Greek Orthodox; 8:Jehovah's Witness; 9:Jewish; 10:Methodist; 11:Muslim; 12:Not Specified; 13:Other; 14:Protestant Quaker; 15:Romanian East; 16:Unobtainable;
Admission Type	1:Elective; 2:Emergency; 3:Urgent;

For our experiments, we suppose there are $M = 4$ hospitals such as H_1, H_2, H_3, H_4.Then, we equally split the dataset into four sub datasets for the 4 hospitals. For our experiment, each plaintext has 32bits. For homomorphic encryption, a 128-bit encryption key is used in our experiment. For the garbled circuit, each key is 64bits.

Suppose each hospital would like to compare its ICU mortality against other 3 hospitals under different age groups. In addition, each hospital may be interested in whether or not its ICU mortality is above the average level among all 4hospitals. Each hospital first computed its local mortality under different age groups, and encrypted these local mortalities with homomorphic encryption, which was sent later to the cloud for secure global comparison. The cloud and the CSP will cooperate together to rank each age group's mortality in each hospital under the proposed framework. As aforementioned, each hospital also wants to know whether or not its mortality is above the average level. To compute the average value, a division circuit under Yao's protocol needs to be implemented, which may significantly increase the circuit complexity. As each hospital is only interested in the comparison, we can reformate the comparison as follows

$$if M \times f_{mor}(D_{H_i}) \geq \sum_{j=1}^{M} f_{mor}(D_{H_i}), \quad (6)$$

Where $M = 4$ is the number of hospitals, D_i is the dataset at hospital H_i and $f_{mor}(D_{H_i})$ is the local function for calculate the local motility. The multiplication and aggregation can be achieved by homomorphic addition and multiplication operations. The results from the garbled circuit are listed in TABLE. In TABLE, the rank of each age group corresponds to the ascend order of mortality, where '1' and '4'refer to the lowest and the highest mortalities, respectively. The mortality of each hospital that is above the average among all hospitals, is shaded in gray in TABLE.

TABLE II: RANKING OUTPUTS USING THE PROPOSED FRAMEWORK, WHERE '1' AND '4' REFERS TO THE LOWEST AND THE HIGHEST MORTALITIES, RESPECTIVELY, WHERE THE MORTALITY THAT IS HIGHER THE AVERAGE AMONG ALL HOSPITALS, IS HIGHLIGHTED IN GRAY.

Age	H_1	H_2	H_3	H_4
20~29	4	1	3	2
30~39	3	4	1	2
40~49	1	2	4	3
50~59	1	4	2	3
60~69	1	2	4	3
70~79	4	1	3	2
>80	3	4	1	2

Where $M = 4$ is the number of hospitals, D_i is the dataset at hospital H_i and $f_{mor}(D_{H_i})$ is the local function for calculate the local motility. The multiplication and aggregation can be achieved by homomorphic addition and multiplication operations. The results from the garbled circuit are listed in TABLE. In TABLE, the rank of each age group corresponds to the ascend order of mortality, where '1' and '4'refer to the lowest and the highest mortalities, respectively. The mortality of each hospital that is above the average among all hospitals, is shaded in gray in TABLE.

TABLE II depicts that the hospitals can obtain useful information for improving their service quality. For example, H_1finds out that the mortality of its young population (age between 20 and 39) is the highest among all hospitals, thus they may need to pay more attention about its young population. This experiment demonstrates a practical use case of the proposed method. In practice, hospitals can conduct more advanced inquiries using the proposed framework. For example, they are able to compare the mortalities of a sub-population. The sub-population can be defined by a criterion like "age is between 50 and 59, sex is male and marriage status is widowed", which can be achieved by modifying the local function in each hospital, whereas the cloud and CSP can follow the same protocols unchanged. Therefore, the proposed framework is flexible to handle more complicated real-world scenarios.

Remark III: For this experiment, we need to design comparison garbled circuit for ranking operation. We designed the comparison circuit as shown in Fig(a), which consists of several full adder circuits(Fig. 6 (b)) and one two's complement circuit (Fig. 6 (a)). In Fig. 6(a), inputs are A and B with L bits, which can represent an integer ranging from 0 to$2^L - 1$, and the output denoted byS_L is the most significant bit of$A - B$, as

$$S_L = \begin{cases} 0, if\ A \geq B \\ 1, otherwise \end{cases}. \quad (7)$$

The comparison circuit can be used to rank the count of records in a query (e.g., a specific combination of attributes).

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

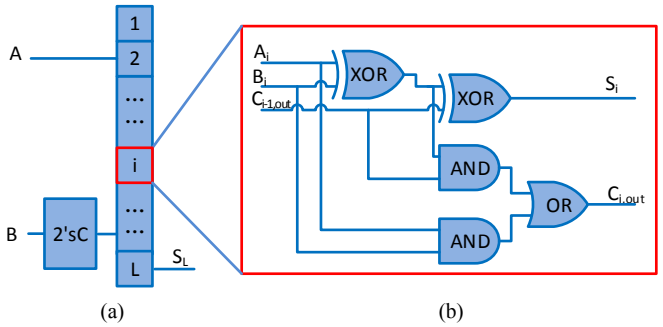


Fig. 6(a) Acomparison circuit, where the comparison is achieved througha full adder circuit as shown in (b) with $C_{0,out} = 0$.

In this section, we used the secure aggregation and comparison operations as examples to illustrate the proposed framework. In practice, the cloud and CSP can securely compute many different functions by combining both homomorphic encryption and garbled circuits. We measured the execution time of some key cryptographic operations in a workstation with an Intel 3.2 GHz CPU, where all the results are averaged over1000 single operations. The execution time of each basic cryptographic primitive has been profiled and shown in TABLEIII.

TABLEIII: THE EXECUTION TIME OF EACH BASIC CRYPTOGRAPHIC PRIMITIVE USED IN THE PROPOSED FRAMEWORK

Operation	Time (milliseconds)
Homomorphic Encryption	3.1
Homomorphic Decryption	4.7
Homomorphic Add	1.7
Homomorphic Multiplication	0.5
OT Protocol	575.0
Comparison Garbled Circuit Evaluation	15.0

IV. Limitations and Discussion

The proposed system has several limitations that need the further investigation. One limitation is that the current framework only supports secure aggregation operations followed by secure rank operations. However, the proposed framework demonstrated the feasibility of combining homomorphic encryption and garbled circuit based crypto techniques for supporting QI studies. In theory, garbled circuit based protocol[1] can be used to securely evaluated arbitrary functions with the cost of specific circuit design. We will leave the circuit design of more advanced functions in our future work.

V. Conclusion

We introduced the PRECISE framework to facilitate privacy-preserving distributed quality metric comparison. The proposed framework supports secure ranking operation on aggregated data from distributed data sources in a cloud-assisted environment. We plan to develop more advanced system such as linear regression system using the garbled circuit protocol and homomorphic encryption primitives in order to expand the usability of the proposed framework.

Acknowledgment

Chen, Wang and Jiang contributed the majority of the writing and conducted major parts of the experiments. Mohammed and Cheng contributed to the methodology and edited the manuscript. This work was funded in part by the NLM (R00LM011392), NLM (R21LM012060), NIGRI (1K99HG008175-01), NHLBI (U54HL108460) and NSERC postdoctoral fellowship.

References

[1] A. C. Yao, “Protocols for secure computations,” in 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982), 1982, pp. 160–164.

[2] Y. Lindell and B. Pinkas, “A proof of security of yao’s protocol for two-party computation,” J. Cryptol., vol. 22, no. 2, pp. 161–188, 2009.

[3] R. Sheikh, B. Kumar, and D. Mishra, “A distributed k-secure sum protocol for secure multi-party computations,” arXiv Prepr. arXiv1003.4071, vol. 2, no. 3, pp. 68–72, 2010.

[4] C. Dwork, “Differential privacy,” Int. Colloq. Autom. Lang. Program., vol. 4052, no. d, pp. 1–12, 2006.

[5] L. Sweeney, “k-anonymity: A model for protecting privacy,” Int. J. Uncertainty, Fuzziness Knowledge-Based Syst., vol. 10, no. 05, pp. 557–570, 2002.

[6] J. Gardner, L. Xiong, Y. Xiao, J. Gao, A. R. Post, X. Jiang, and L. Ohno-Machado, “SHARE: system design and case studies for statistical health information release.,” J. Am. Med. Inform. Assoc., vol. 20, no. 1, pp. 109–16, Jan. 2013.

[7] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, “L-diversity: privacy beyond k-anonymity,” in Proceedings of the 22nd International Conference on Data Engineering, 2006, pp. 1–12.

[8] N. Li, T. Li, and S. Venkatasubramanian, “t Closeness : Privacy Beyond k-Anonymity and -Diversity,” in Data Engineering, IEEE 23rd International Conference on, 2007, no. 2, pp. 106–115.

[9] C. Dwork, F. McSherry, K. Nissim, and A. Smith, “Calibrating noise to sensitivity in private data analysis,” Theory Cryptogr., vol. 3876, no. 1, pp. 265–284, 2006.

[10] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, “Order preserving encryption for numeric data,” in Proceedings of the 2004 ACM SIGMOD international conference on Management of data - SIGMOD ’04, 2004, p. 563.

[11] M. Naehrig, K. Lauter, and V. Vaikuntanathan, “Can homomorphic encryption be practical?,” in Proceedings of the 3rd ACM workshop on Cloud computing security workshop - CCSW ’11, 2011, p. 113.

[12] M. Ogburn, C. Turner, and P. Dahal, “Homomorphic encryption,” in Procedia Computer Science, 2013, vol. 20, pp. 502–509.

[13] C. Fontaine and F. Galand, “A survey of homomorphic encryption for nonspecialists,” EURASIP J. Inf. …, 2007.

[14] K. Gjøsteen, “A New Security Proof for Damgard’s ElGamal,” in Topics in Cryptology – CT-RSA, 2006, pp. 150–158.

[15] D. Boneh and H. Shacham, “Fast variants of RSA,” CryptoBytes, pp. 1–10, 2002.

[16] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in Advances in cryptology—EUROCRYPT’99, 1999, pp. 223–238.

[17] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) fully homomorphic encryption without bootstrapping,” in Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS ’12, 2012, vol. 111, no. 111, pp. 309–325.

[18] C. Gentry and S. Halevi, “Implementing gentry’s fully-homomorphic encryption scheme,” Adv. Cryptology–EUROCRYPT 2011, 2011.

978-1-4799-7294-4/14/$31.00 ©2014 IEEE

[19] C. Gentry, "A fully homomorphic encryption scheme," Stanford University, 2009.
[20] M. Van Dijk and C. Gentry, "Fully homomorphic encryption over the integers," Adv. Cryptology– …, 2010.
[21] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," Found. Comput. …, 2011.
[22] S. Even, O. Goldreich, and A. Lempel, "A randomized protocol for signing contracts," Commun. ACM, vol. 28, no. 6, pp. 637–647, Jun. 1985.
[23] J. D. Golić, "Cryptanalysis of alleged A5 stream cipher," in Advances in Cryptology—EUROCRYPT'97, 1997, pp. 239–255.
[24] "Announcing the Advanced Encryption Standard (AES)," Fed. Inf. Process. Stand. Publ., 2001.
[25] "MIMIC II Clinical Database." [Online]. Available: https://mimic.physionet.org/database.html.

978-1-4799-7294-4/14/$31.00 ©2014 IEEE