Privacy preserving health data processing

Anders Andersen
Department of Computer Science
Faculty of Science and Technology
UiT The Arcic University of Norway
9037 Tromsø, Norway
Email: Anders.Andersen@uit.no

Kassaye Yitbarek Yigzaw
Department of Computer Science
Faculty of Science and Technology
UiT The Arcic University of Norway
9037 Tromsø, Norway
Email: Kassaye.Y.Yigzaw@uit.no

Randi Karlsen
Department of Computer Science
Faculty of Science and Technology
UiT The Arcic University of Norway
9037 Tromsø, Norway
Email: Randi.Karlsen@uit.no

***Abstract*—The usage of electronic health data from different sources for statistical analysis requires a toolset where the legal, security and privacy concerns have been taken into consideration. The health data are typically located at different general practices and hospitals. The data analysis consists of local processing at these locations, and the locations become nodes in a computing graph. To support the legal, security and privacy concerns, the proposed toolset for statistical analysis of health data uses a combination of secure multi-party computation (SMC) algorithms, symmetric and public key encryption, and public key infrastructure (PKI) with certificates and a certificate authority (CA). The proposed toolset should cover a wide range of data analysis with different data distributions. To achieve this, large set of possible SMC algorithms and computing graphs have to be supported.**

I. Introduction

In this paper an approach for analyzing health data from different general practices and hospitals that take into consideration legal, security and privacy issues is presented and discussed. The approach combines secure multi-party computation (SMC) algorithms [1], [2] with symmetric and public key encryption, public key infrastructure (PKI), certificates, and a certificate authority (CA).

The work is done in the context of the Snow SMSC (Secure Multi-party Statistical Computations) project. Snow SMSC is based on earlier work on the Snow system [3], which is used in data extraction for disease surveillance. In the Snow SMSC project health data from health institutions in Norway are aggregated and analyzed.

An example of a situation where the patient's privacy should not be infringed is the task of analyzing the usage of certain medications at all health institutions in a region based on sparse data. When collecting statistical data for this task, an unwanted effect could be the ability to identify the most likely patient using some specific sort of medication. This would be the case if the data comes from one health institution (e.g. a general practice) with few patients matching the typical user of such medications. For example if the data shows that only one patient at that general practice is subscribed medication a that is typical for multiple sclerosis (MS) patients, and only one patient at that general practice has visible symptoms of MS. When such data is part of a computation or analyzing task, special care has to be taken to ensure the privacy of the patients.

II. Requirements

The outcome of SMC research [4] will in combination with cryptography be used to address legal, security and privacy issues. The constraints for SMC discussed in [5] combined with a practical and efficient implementation [6] are the basis for our work. Secure disease surveillance might enforce other requirements [7] that should be considered in future work.

In [5], SMC for N institutions with different data sets who wish to evaluate a result r (e.g. the correlation of usage of medication x and the adverse effect y) based on the data sets is subject to four constrains:

- C1: The correct value result r is obtained and known to all institutions.
- C2: No institution learns more about the other institutions values than it can deduce from its own data set and the result r.
- C3: No trusted third party—human or machine—is part of the process.
- C4: *Semi-honesty*. Institutions perform agreed upon computations correctly using their true data. However, they are permitted to retain the results of intermediate computations.

For practical purposes, the zero information disclosure implied by C3 is sometimes difficult to implement efficiently [6]. In the context of the Snow SMSC project we will use the following relaxed constrain replacing C3:

- C3': No human or machine is allowed to have access to both the patient identifier, and data of that patient not previously known.

To fulfill the requirements, a combination of SMC algorithms and careful usage of encryption and certificates are used. The approach is based on a coordinator that prepares the computation and a set of sub-processes (nodes) representing the parties in the multi-party computation. In this paper the focus is the combined usage of SMC algorithms and cryptography to achieve privacy.

In [8] the privacy concerns and how this is implemented is discussed, and in [9] the focus is how to use SMC to process distributed health data. In [10] a complete architecture for SMC in healthcare environments is discussed.

978-1-4799-6644-8/14/$31.00 ©2014 IEEE

III. Secure Multi-Party Computations

Secure multi-party computation (SMC) deals with the problem of a set of nodes $P = \{n_1, n_2, \ldots, n_m\}$ jointly computing an arbitrary function $f(\ldots)$ on their private input, while ensuring security properties, such as input privacy and correctness of output. The security properties should be preserved even in the face of an adversary that controls a subset of the institutions and wish to attack the computation [11].

The first step to preserve privacy is to choose an algorithm that matches the four constrains presented above. SMC algorithms ensure that each participant performing a sub-process is unable to learn about the other participants input data and intermediate results.

For example, to calculate the mean value of N numbers from N participants, the participants can be ordered in a chain where each participant receives a value from the previous participant in the chain, adds its number to the value, and forwards the new value to the next participant in the chain. If the first participant in the chain starts with the value 0 and adds its number to the value, the last participant can calculate the mean value by adding its number to the value received and dividing the value by N. If we add a coordinator to the beginning and the end of the chain, the coordinator can send the value 0 to the first participant and calculate the mean value at the end of the chain. Figure 1 illustrates this chain (computation graph) where n_c is the coordinator and n_1, n_2, and n_3 are the participants. r_0 is the initial value 0, x_i is the number of participant n_i, r_i is the intermediate results from participant n_i, $\{r_i\}$ is the message from participant n_i to participant n_{i+1}, and m is the calculated mean value.

This solution does not preserve privacy. For example, the value participant 2 receives is equal to the number x_1 of participant n_1. To preserve privacy the participants should not be able to deduce the private numbers' of the other participants. In an SMC version of this algorithm that preserves the privacy of the participants, step ⓪ and ④ are replaced with the following versions (*rand*() generates a large unique random number):

$$⓪ \quad \boxed{r_0 = \mathit{rand}()}$$

$$④ \quad \boxed{m = \frac{r_3 - r_0}{3}}$$

Since r_0 is a large random number unknown to the participants, there is no way for them to deduce the other participants' numbers from the values (intermediate results) they receives.

In Figure 1, a computation graph with 4 nodes n_c, n_1, n_2, and n_3 is shown. The figure includes the nodes, the messages (the edges of the directed graph labeled with the content of the messages inside curly braces), and the processing at each node (inside rectangular boxes labeled to order them). Such graph representation can be used to illustrate a large number of SMC algorithms. In [8] a syntax to specify such graphs is presented. This graph representation is used at the coordinator to construct the computation graph and distribute this information to the nodes. We will later see that at each node only information about the neighbour nodes in the graph is revealed.

TABLE I. The notation used for encryption and signing.

$\{m\}$	A message containing m
$s\{m\}$	m encrypted with secret key s
$\{m\}_p$	m encrypted with public key p
$\{m\}^a$	m signed by a
$\{m\}^a_p$	m signed by a and encrypted with public key p
$\{a, p\}^c$	CA c binds public key p to identity (address) a
$A \rightarrow B : \{m\}$	Message $\{m\}$ sent from A to B

IV. Message encryption

To make the SMC example above privacy preserving, the messages have to be encrypted in such a way that only the intended receiver can read its content. For example, if an attacker is able to read the content of message $\{r_0\}$ and message $\{r_1\}$, the number x_1 of node n_1 is exposed ($x_1 = r_1 - r_0$). The coordinator is responsible for constructing the computation graph, and as part of this process provide the necessary information for the nodes to be able to properly encrypt the messages they create and forward.

Each node has its own public/private key pair. These keys are used to encrypt and decrypt the messages to the node. The content of each message is actually encrypted using symmetric encryption. The secret key of the symmetric encryption is fresh and unique for this message. The secret key of the message is encrypted with the public key of the intended receiver and included in the message. If s is the secret key of a message with content m and the intended receiver has the public key p then this is the complete encrypted message:

$$\Big\{\{s\}_p, s\{m\}\Big\}$$

The notation[1] used is described in Table I. The intended receiver first decrypts $\{s\}_p$ using its private key. It can then use s to decrypt $s\{m\}$ to be able to access m. Since only the intended receiver has access to its private key, the content m of the message is only accessible to that node. The usage of symmetric encryption is an implementation detail. To simplify the notation in the following text, messages like the one above will be denoted like this:

$$\{m\}_p$$

For a node to be able to create encrypted messages to the next nodes in the computation graph, the message it receives has to include the addresses and public keys of these nodes. For each of the next nodes in the computation graph it receives the address n, the public key p, and an information blob b. The information blob b is encrypted with the receiving nodes's public key at it is therefore not readable by the current node. It should be forwarded to the receiving node unmodified. The current node receives a similar information blob encrypted by its public key. This information blob contains the addresses, public keys, and information blobs of the next nodes in the

[1]The notation used in this paper is inspired by the notion found in [12], [13], and [14]. In [13] square brackets are used for encryption and decryption using the private key in a public key crypto system. The notation $\{m\}^a$ used in this paper to sign a message could with the square bracket operation with private key notation be equal to $\{m, [h(m)]_a\}$, where h is a cryptographic hash function.

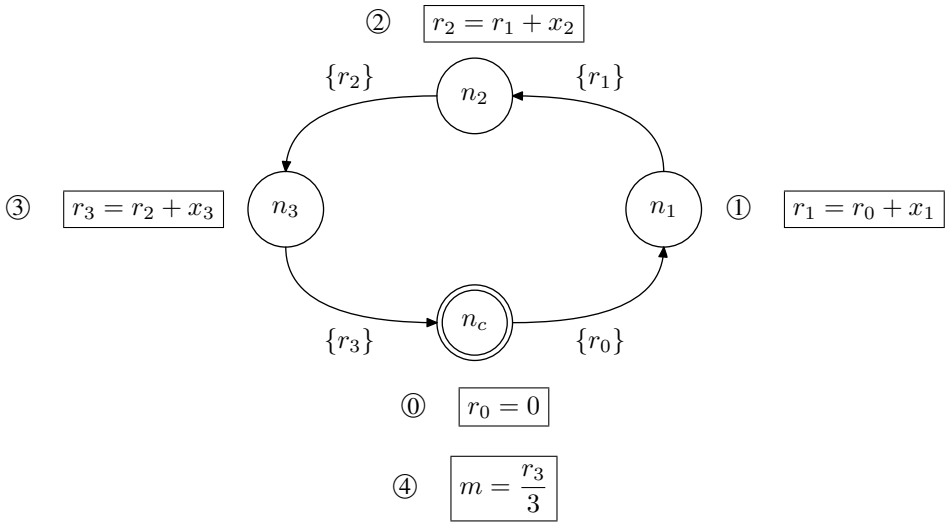

Fig. 1. Calculating the mean value m with three participants $n1$, $n2$, and $n3$ and a coordinator n_c.

computation graph. Using this approach, the message between node n_1 and n_2 in Figure 1 should be the following:

$$n_1 \rightarrow n_2 : \Big\{\{r_1\}_{p_2}, b_2\Big\}, \quad \text{where } b_2 = \{n_3, p_3, b_3\}_{p_2}$$

The first part of the message $\{r_1\}_{p_2}$ is generated by node n_1 using the public key p_2 of node n_2 received in the information blob b_1 in the message to node n_1. The second part of the message b_2 was generated by the coordinator and is forwarded by node n_1 without modification. Since this blob was encrypted with the public key of node n_2 at the coordinator, node n_1 have no way to access its content. When node n_2 receives the message it can use its private key to decrypt both part of the message. It can then get access to the intermediate result r_1 from node n_1 and the information about the next nodes found in b_2. This information is then used to perform the processing at node n_2, and create and forward the message to node n_3.

V. PKI AND CERTIFICATES

Using public-key encryption will protect the content of the messages intended for a given node. The problem is that the receiver can not trust the content. Anyone with access to the public key of a node can create a fake message that can be used to get access to data from this node. For example, if node n_1 in Figure 1 is compromised, it can create the following message for node n_2:

$$n_1 \rightarrow n_2 : \Big\{\{y\}_{p_2}, b_2\Big\}, \quad \text{where } b_2 = \{n_1, p_1, \ldots\}_{p_2}$$

In this attack the second part of the message b_2 was created by the compromised node n_1 and not by the coordinator. The compromised node inserted its own address and public key into b_2. The consequence is that node n_2 will perform its calculation and return the result to node n_1. The compromised node n_1 can then calculate the number x_2 of node n_2:

$$x_2 = r_2 - y$$

It is also possible for unknown nodes to generate such messages to reveal data from a node. This demonstrates that authenticity and integrity is important in a system like this. The receiver of a message has to be assured that the message is received from an expected node in the computation and that the message has not been altered.

To achieve this a public key infrastructure (PKI) with digital certificates are introduced. We will in this paper not describe a PKI in detail but focus on how the certificates provided and managed by the PKI can be used to ensure the privacy in Snow SMSC. A certificate is a digital signed document that binds a public key to an identity. This is important for digital signatures. If a document (or some data) is signed, the signature can be validated by the public key of the signee. If this public key came from a certificate, the signature can be verified towards the identity. We can trust the certificates since they are signed by a certificate authority (CA). Conceptually, we can think of the certificate of a node as a message signed by the CA that contains both the address (identity) and the public key of the node. In this example the CA c binds address n_i to public key p_i.

$$\{n_i, p_i\}^c$$

To be able to verify the certificates of the nodes, we will assume that the certificate of the CA is preinstalled on all nodes and that the CA is trusted.

The important part of introducing PKI and certificates is that we can sign and validate the information exchanged in the messages in Snow SMSC. If information that is digitally signed is modified the signature will not match the information and we can not verify that the information originated from the claimed sender. The signature is verified before the message is interpreted. $\{m\}^a$ is the notation used for a message $\{m\}$ that is signed by a and can be verified by the public key p_a of a. A notation for a message that is both signed by a sender and encrypted with the public key of the receiver also exists. The following message $\{m\}$ is encrypted with the public key p_u and its signature can be verified by public key of v:

$$\{m\}^v_{p_u}$$

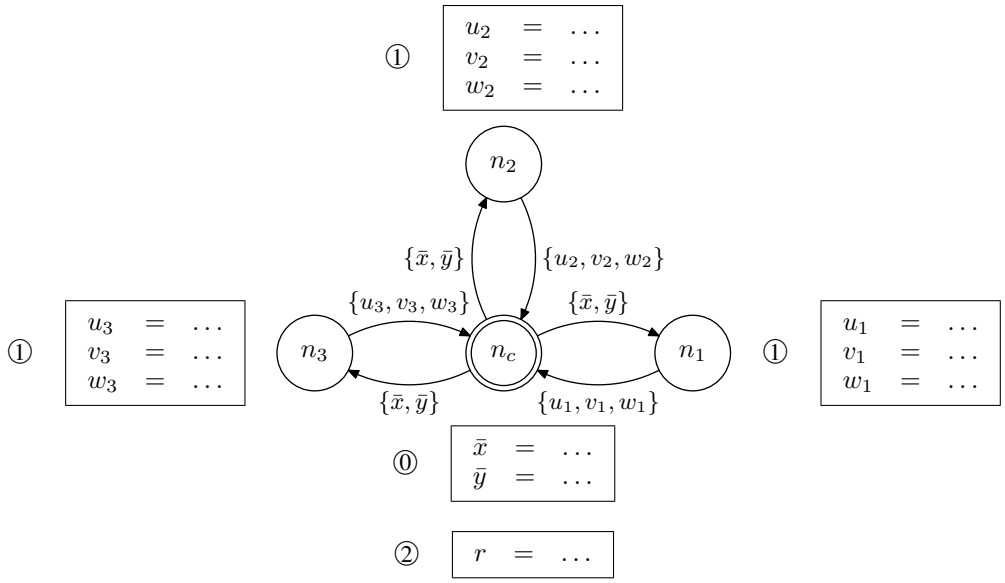

Fig. 2. Calculating Pearson's r of n samples of two variables x and y at 3 health institutions.

The signature of a message is verified to check the authenticity and integrity of the message. It is also important to verify that a message is part of the computation and that it was the intention of the coordinator that this message should be sent from the sender to the receiver. In other words, the message is part of the planned computation graph. So far we have ignored that this information should be included in the information blob generated by the coordinator for the current node. The actual message from node n_1 to node n_2 in the example in Figure 1 with encryption, signatures and the information about the intended sender be like this:

$$n_1 \rightarrow n_2 : \Big\{\{r_1\}_{p_2}, b_2\Big\}^{n_1}$$

Where:

$$b_2 = \Big\{\{n_1, p_1\}^c, \{n_3, p_3\}^c, b_3\Big\}_{p_2}^{n_c}$$

The complete message is signed by node n_1. The information blob b_2 is signed by the coordinator n_c and encrypted with p_2, the public key of node n_2. Since b_2 was generated and encrypted at the coordinator we can be ensured that the information found here include the intentions of the coordinator. The first element in b_2 is the coordinator's information to n_2 about who this message should be received from. Node n_2 should verify the signature of the complete message towards the information in this certificate. The rest of the elements in b_2 is concerned with the following nodes in the computation graph. The second element is the certificate that contains the address and the public key of node n_3. The last element is the information blob b_3 that n_2 is going to forward unmodified to node n_3. The trust we can establish from this is the following:

(i) The complete message originates from node n_1 since it is signed by n_1.

(ii) The message was expected to come from node n_1 since the first element of b_2 is the certificate of node n_1 signed by CA c.

(iii) The information blob b_2 originates from the coordinator since it is signed by the coordinator n_c.

(iv) The information blob b_2 is created for n_2 since it is encrypted with its public key p_2. It can also be concluded that this was the coordinator's intention since the encrypted b_2 is signed by the coordinator.

Every message and information blob in Snow SMSC are encrypted and signed as illustrated in the example above. This ensures the authenticity and integrity of the messages and their content.

VI. An example

The calculation of the Pearson product-moment correlation coefficient (Pearson's r) on horizontally partitioned dataset[2] is used as an example. The actual calculation is not important for the example, but it is included for completeness. The computation graph in Figure 2 shows the nodes, the messages and the processing needed to follow the example. Pearson's r is used to measure the correlation (linear dependence) between n samples of two variables x and y:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

In the case of m health institutions with s_j samples of x_{j_i} and y_{j_i} at each institution, r can be rewritten like this:

$$r = \frac{\sum_{j=1}^{m}\sum_{i=1}^{s_j}(x_{j_i} - \bar{x})(y_{j_i} - \bar{y})}{\sqrt{\sum_{j=1}^{m}\sum_{i=1}^{s_j}(x_{j_i} - \bar{x})^2 \sum_{j=1}^{m}\sum_{i=1}^{s_j}(y_{j_i} - \bar{y})^2}}$$

At each node j the following three intermediate results have to be calculated:

① $\quad u_j = \sum_{i=1}^{s_j}(x_{j_i} - \bar{x})(y_{j_i} - \bar{y})$
$\quad\quad v_j = \sum_{i=1}^{s_j}(x_{j_i} - \bar{x})^2$
$\quad\quad w_j = \sum_{i=1}^{s_j}(y_{j_i} - \bar{y})^2$

[2]In a horizontally partitioned dataset each institution has all variables of different entities.

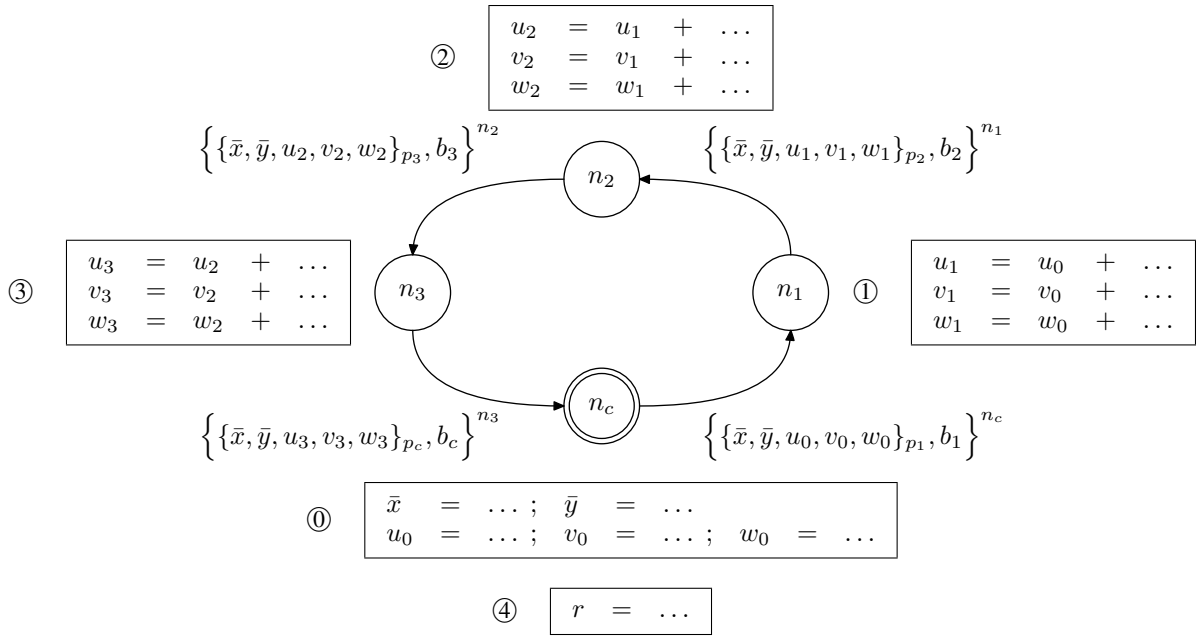

Fig. 3. Calculating Pearson's r of n samples of two variables x and y at 3 health institutions without exposing the intermediate results.

The initial mean values $\bar{x}$ and $\bar{y}$ can be securely calculated ⓪ using an approach similar to the SMC example calculating the mean value in Section III. When all intermediate results are received at the coordinator, Pearson's r can be calculated:

② $$r = \frac{\sum_{j=1}^{m} u_j}{\sqrt{\sum_{j=1}^{m} v_j \sum_{j=1}^{m} w_j}}$$

Figure 2 illustrates the computation graph for this problem with 3 health institutions before encryption, digital signatures and certificates are introduced.

The messages between the coordinator and the health institutions will be similar. For any health institution n_j the message from the coordinator n_c to node n_j will be the following when encryption, digital signatures and certificates are introduced:

$$n_c \rightarrow n_j : \left\{\{\bar{x}, \bar{y}\}_{p_j}, b_j\right\}^{n_c},$$

where

$$b_j = \left\{\{n_c, p_c\}^c, \{n_c, p_c\}^c, b_c\right\}_{p_j}^{n_c}$$

The interesting part here is the content of b_j. The first element tells node n_j that it was the intention that this message should arrive from n_c. The second element tells node n_j that the next node in the computation graph is n_c. After node n_j has calculated u_j, v_j and w_j, it generates and forwards the following message to n_c:

$$n_j \rightarrow n_c : \left\{\{u_j, v_j, w_j\}_{p_c}, b_c\right\}^{n_j},$$

where

$$b_c = \left\{\{n_j, p_j\}^c\right\}_{p_c}^{n_c}$$

The information blob b_c only contains one element since there are no next nodes in the computation graph. The coordinator uses b_c to verify that this message was intended for the coordinator before it collects the data. After the coordinator has received similar messages from all health institutions it uses the data to calculate the Pearson's r for x and y.

The argument for privacy preserving in this example is similar to the one in the previous example. However, if the intermediate results u_j, v_j and w_j are considered private it is easy to see from the computation graph in Figure 2 and processing step ② that the values will be exposed at the coordinator n_c. This is solvable since the first thing the coordinator has to do in step ② is to summarize all the values received from the nodes. Instead of sending these values to the coordinator directly, a similar approach to the one done when calculating the mean value in the example from Figure 1 can be selected. First we generate three large unique random numbers u_0, v_0, and w_0 at the coordinator ⓪. These numbers are sent, together with the mean values $\bar{x}$ and $\bar{y}$, as intermediate results to the first node n_1. At each node n_i the following calculations are performed (①, ②, and ③):

$$\begin{aligned} u_j &= u_{j-1} + \sum_{i=1}^{s_j} (x_{j_i} - \bar{x})(y_{j_i} - \bar{y}) \\ v_j &= v_{j-1} + \sum_{i=1}^{s_j} (x_{j_i} - \bar{x})^2 \\ w_j &= w_{j-1} + \sum_{i=1}^{s_j} (y_{j_i} - \bar{y})^2 \end{aligned}$$

When the calculations are performed, node n_i generates and forwards the following message to node n_{i+1}:

$$n_i \rightarrow n_{i+1} : \left\{\{\bar{x}, \bar{y}, u_i, v_i, w_i\}_{p_{i+1}}, b_{i+1}\right\}^{n_i}$$

Finally, the coordinator n_c receives the values u_3, v_3, and w_3. To get the real value used to calculate Pearson's r, the random

numbers u_0, v_0, and w_0 has to subtracted from u_3, v_3, and w_3, respectively. The coordinator calculates Pearson's r ④:

$$④ \quad \boxed{r = \frac{u_3 - u_0}{\sqrt{(v_3 - v_0)(w_3 - w_0)}}}$$

VII. Evaluation

Semi-honest adversary is one of the commonly considered adversarial behaviors in SMC, where each node is assumed to follow the protocol, but they might try to extract privacy sensitive information from the messages exchanged during the computation. The secure summation protocol used in this article to calculate mean values also assumes a modified form of semi-honest adversary. Semi-honest adversarial behavior can be acceptable among healthcare institutions where the institutions can be trusted to follow the protocol.

Unlike the conventional semi-honest adversary model, the protocol has a mechanism to enforce the execution of the protocol in the order specified by the computation graph. The graph contains a message (blob) to each node, signed by the coordinator, that specifies the previous and next nodes in the computation. The graph can not be modified without invalidating the signature. Therefore, the protocol can only be executed in the order specified by the coordinator.

If any node uses different values than its true values, the protocol cannot ensure correctness of output. The protocol assumes that the nodes use their true values in the computation.

The protocol can be vulnerable for collision attack by two nodes. For instance, in Figure 1, if node n_1 and node n_3 collide they will know the value of node n_2. The protocol assumes any two nodes will not collide. However, since the computing graph is generated by a coordinator, computing nodes only have knowledge about their predecessor and successor nodes that makes collusion difficult. In addition, if a coordinator node creates a computing graph in a way the next node is itself, it is possible to learn other node's value. The protocol is privacy preserving on the assumption that a coordinator generates the correct graph. These kinds of assumptions can be acceptable among health institutions.

VIII. Conclusion

It has been demonstrated that a combination of SMC, encryption and PKI can be used to perform privacy-preserving statistical analyses on distributed health data. A layered structure of the computation graph has been proposed, where each layer is encrypted and signed and can be revealed and verified one layer at the time through the computation steps.

A coordinator specifies the secure protocols as a computation graph. Each step in the computation is represented as layers, where the first layer is the outermost layer. At each node the one layer can be revealed using the private key of the node. The next layers are inaccessible since they are protected by the next nodes private keys.

The computation graph is included in the messages between the nodes. The graph is revealed step by step in the process of performing the distributed computation. In a node the accessible layer of the graph is used to verify both that the originator of the received message and the next nodes were the intention of the coordinator. The signed intermediate results received can, by verifying the signature, be verified to originate from the sender of the received message (that is already verified to be the intended sender by coordinator).

The example in Section VI demonstrates the computation of Pearson's r based on the idea of decomposing an equation into a set of sub-computations, which are in summation form, and each sub-computation is computed using a secure protocol. Similarly, a large number of linear and non-linear statistical problems can be decomposed into a set of sub-computations of summation forms [15]. Therefore, the method described in the paper can be applied for a large number of statistical computations.

References

[1] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game, or a completeness theorem for protocols with honest majority," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. New York: ACM, 1987, pp. 218–229.

[2] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. New York: ACM Press, 1982, pp. 160–164.

[3] J. G. Bellika, G. Aronsen, M. A. Johansen, G. Hartvigsen, and G. S. Simonsen, "The snow agent system: A peer-to-peer system for disease surveillance and diagnostic assistance," *Advances in Disease Surveillance,*, vol. 4, p. 42, 2007.

[4] S. Goldwasser, "Multi party computations: past and present," in *PODC'97, Proceedings of the sixteenth annual ACM symposium on Principles of distributed computing*. New York: ACM, 1997, pp. 1–6.

[5] A. F. Karr, "Secure statistical analysis of distributed databases, emphasizing what we don't know," *Journal of Privacy and Confidentiality*, vol. 1, no. 2, pp. 197–211, 2009.

[6] W. Du and Z. Zhan, "A practical approach to solve secure multi-party computation problems," in *Proceedings of the 2002 workshop on New security paradigms*. New York: ACM, 2002, pp. 127–135.

[7] K. E. Emam, J. Hu, J. Mercer, L. Peyton, M. Kantarcioglu, B. Malin, D. Buckeridge, S. Samet, and C. Earle, "A secure protocol for protecting the identity of providers when disclosing data for disease surveillance," *J Am Med Inform Assoc*, vol. 18, pp. 212–217, 2011.

[8] A. Andersen, "An implementation of secure multi-party computations to preserve privacy when processing EMR data," in *The International Conference on Privacy, Security and Trust (PST 2013)*, 2013.

[9] ——, "Using secure multi-party computation when pocessing distributed health data," in *The 2013 International Conference on Security and Management (SAM'13)*, 2013.

[10] K. Y. Yigsaw, J. G. Bellika, G. Hartvigsen, and A. Andersen, "Towards a privacy preserving computation on distributed health records," in *Middleware Doctoral Symposium 2013*, 2013.

[11] Y. Lindell and B. Pinkas, "Secure multiparty computation for privacy-preserving data mining," *Journal of Privacy and Confidentiality*, vol. 1, no. 1, pp. 59–98, 2009.

[12] C. Kaufman, R. Perlman, and M. Speciner, *Network Security: Private Communication in a Public World*, 2nd ed. Prentice Hall, 2002.

[13] M. Stamp, *Information Security: Principles and Practice*. John Wiley & Sons, 2012.

[14] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Transactions on Computer Systems*, vol. 8, pp. 18–36, 1990.

[15] C. T. Chu, S. K. Kim, Y. A. Lin, Y. Yu, G. R. Bradski, A. Y. Ng, and K. Olukotun, "Map-reduce for machine learning on multicore," in *NIPS*. The MIT Press, 2006, pp. 281–288.