A Peer-to-Peer Protocol and System Architecture for Privacy-Preserving Statistical Analysis

Katerina Zamani, Angelos Charalambidis, Stasinos Konstantopoulos,
Maria Dagioglou, and Vangelis Karkaletsis

Institute and Informatics and Telecommunications,
NCSR ‘Demokritos’,
Ag. Paraskevi, Greece
{kzam,acharal,konstant,mdagiogl,vangelis}@iit.demokritos.gr

Abstract. The insights gained by the large-scale analysis of health-related data can have an enormous impact in public health and medical research, but access to such personal and sensitive data poses serious privacy implications for the data provider and a heavy data security and administrative burden on the data consumer. In this paper we present an architecture that fills the gap between the statistical tools ubiquitously used in medical research on the one hand, and privacy-preserving data mining methods on the other. This architecture foresees the primitive instructions needed to re-implement the elementary statistical methods so that they only access data via a privacy-preserving protocol. The advantage is that more complex analysis and visualisation tools that are built upon these elementary methods can remain unaffected. Furthermore, we introduce RASSP, a secure summation protocol that implements the primitive instructions foreseen by the architecture. An open-source reference implementation of this architecture is provided for the R language. We use these results to argue that the tension between medical research and privacy requirements can be technically alleviated and we outline a research plan towards a system that covers further requirements on computation efficiency and on the trust that the medical researcher can place on the statistical results obtained by it.

Keywords: Privacy-Preserving statistical analysis; Secure summation protocol; Statistical processing of health records.

1 Introduction

The insights gained by the large-scale analysis of health-related data can have an enormous impact in public health and medical research, but access to such personal and sensitive data poses serious privacy implications for the data provider and a heavy data security and administrative burden on the data consumer. The discussion on what exactly it means to not disclose private data [4] and the discussion on policies for balancing between scientific advancement and privacy [7] are very relevant, but should be complemented by the equally relevant discussion of whether there is tension at all between data privacy and data-driven research. In other words, it is not straightforward if private data can be insulated from medical research workflows without compromising either.

As anonymization has been repeatedly proven to be inadequate [15], attention has turned to research in cryptography and distributed computation. These fields can provide methods for computing aggregates and statistics without revealing the specific data values involved in the computation, offering a much stronger guarantee of privacy than anonymization. However, from the perspective of the data mining practitioners and the medical researchers there is still a residue of functionality missing between their workflows over anonymized data and what is technically possible to achieve without accessing specific datapoints. Naturally, part of the workflow involves browsing data in order to formulate a hypothesis, and cannot possibly be performed over anything else but experimental data specifically collected and licensed to be shared. The scope of our discussion is, therefore, necessarily restricted to the data and processing required to empirically validate an already formulated hypothesis over a larger dataset than what can reasonably be made available to research.

To make this more concrete, we will assume use cases from *ambient assisted living (AAL)* environments. AAL covers a wide range of concepts, hardware and software products, and services that facilitate better, healthier and safer life outside formal health-care institutions. These environments emphasise the automatic collection of health data in one's own environment and the secure sharing of such data with medical care providers. In such a system, health data is shared between the following entities:

- The *AAL agent* that is the data management component of the AAL environment. The AAL agent has unrestricted access to its user's sensitive data. The management and security of the data held by the AAL agent is primarily within the scope of network security.
- The *health-care provider* that needs access to sensitive data of a small set of individuals on a need-to-know basis, depending on the medical condition that necessitates the monitoring of each individual. The management and security of the data held by the health-care provider is primarily within the scope of network security and access control.
- The *medical researcher* that needs access to aggregate values computed over the sensitive data of a large set of individuals, but does not need to know any specific individual's data. It is the data transfer protocols between this agent and the AAL agents that are within the scope of the work described here.

In the remainder of this paper, we first present the main approaches to privacy-preserving computation and discuss what requirements from our use case are not covered by the state of the art (Section 2). We then proceed to present a system architecture that exposes privacy-preserving computation functionality to tools (such as R) that are commonly used in medical research workflows (Section 3). We then present our peer-to-peer protocol that implements this privacy-preserving computation functionality based on the homomorphic property of composite secret sharing schemes (Section 4). We finally conclude and discuss future research direction (Section 5).

2 Related Work

We see in the literature three major approaches to privacy-preserving computation: *differential privacy*, *homomorphic encryption*, and *secure multiparty computation*. *Differential privacy* is based on the property that a result of a statistical value can be approximated even if random noise has been added to the data. *Homomorphic encryption* supports computations over cipher-texts, so that the result can be obtained without decrypting individual datapoints. Finally, *secure multiparty computation* is based on communication protocols between the agents to collaboratively compute a function over their private values without revealing the actual values.

Differential privacy preserves privacy by perturbing the datasets with randomized noise, such as symmetric exponential (Laplace) noise or with a use of a Geometric Distribution [18]. When the perturbed datasets are used in statistical analysis, knowledge of the distribution parameters of the noise applied allows approximating the analysis outcomes over the unperturbed data, but does not allow recovering any of the individual datapoints. To name an example, the PINQ data analysis platform [13] creates a differential privacy layer between the raw data and data analysis software. PINQ supplies the analyst with a set of transformations in operations like Where, Select, GroupBy and Join, in order to apply them to the data-set before applying operations for differential-privacy aggregations.

What should be noted about differential privacy is that it provides approximations and is only applicable where this is tolerated and where the datasets are large enough to allow for this approximation to be accurate enough for its purpose. In the the analysis for medical data, it is often the case that datasets are not large enough to give tolerable error margins or that outliers can lead to important insights and should be highlighted rather than smoothed out.

The second major strain of privacy-aware computation protocols is based on *homomorphic cryptosystems*, cryptographic mechanisms with the property that certain operators (such as addition) can be computed directly within the encrypted space without requiring that the individual operands can be decrypted. One of the most prominent homomorphic cryptosystems is Paillier's cryptosystem [16], which allows computing the cipher of the sum of two numbers given the ciphers of these numbers. Paillier's cryptosystem requires that all numbers are encrypted using derivatives of the public part of a master key; these derivatives are such that they cannot decrypt the cipher of other derivative keys, but the master key can decrypt the cipher of the sum. This algorithmic basis can be extended to provide further numerical and categorical operators beyond summation; for example Kissner and Song [12] proposed an extension that supports union, intersection and element reduction.

Although data providers are perfectly protected from their peers, the main weakness of homomorphic systems is the trust that must be placed on the entity that issues the master key [10]. The typical summation protocol based on Paillier's cryptosystem has a master agent issue a master key and a number of data agents that exchange their encrypted values between them in order to send a total encrypted summation back to the master agent. Privacy from the master agent is only guaranteed by the fact the master agent only receives the cipher of the end-result. If the master agent colludes with one malicious data agent, they can use the private part of the master key to reveal the private value of the victim agent, the data agent that passes its encrypted data to the malicious agent.

To lift the requirement that the master agent must be trusted, Shi et al. [18] proposed a framework that can compute statistics on medical data with the use of an *untrusted* data aggregator, by encrypting values that can be decrypted with the sum of multiple cipher-texts under different user keys. Shi et al. propose a method where each agent encrypts periodically its data with its respective private key. The data of every agent includes its private value combined with white noise. The untrusted aggregator receives all the encrypted values from the agents and decrypts them with its private key and with the use of a correlation between the private keys of all agents and a specific hash function, that is based on the time series. The algorithm needs an initial trusted setup phase, which does not allow agents to join or leave the system dynamically. The proposed protocol is based on differential privacy and as an implication the resulted statistic is an approximation of the real one, which may cause problems in medical data. Moreover, as authors report, in order for their approach to work efficiently, the plain text space should be small.

There are many studies that combine their secure mechanism with the use of a trusted third party that works as the aggregator. In trusted third party protocols, there is an external trusted party which receives the private data of the agents and computes a function by using them. Hanmanthu et al. [5] propose an enhanced protocol that combines a technique which perturbs distributed data with the use of a third party. Specifically, they define a protocol for constructing a Naive Bayes classifier. In this protocol, each agent encrypts its perturbed data with its private key and sends it to a trusted third party. The trusted third party decrypts this data with the public key of the respective agent and constructs a perturbed Naive Bayes Classifier. Moreover, there are some studies that combine *secure multiparty computation (SMC)* systems with a trusted third party. Generally speaking, an SMC system deals with the computation of any function with any input in a distributed network, where the involved agents can learn only the total result and their own input. Thus, a common strategy to ensure trustworthiness is the use of a trusted third party. Ajmani et al. [1] present TEP, a trusted third party computation service that maintains generality. TEP offers flexibility because it fits in many SMC applications to guarantee privacy. However, this type of mechanism requires the existence of a trusted third party, so is inherently weaker than purely peer-to-peer networks.

Nevertheless, Sheikh et al. [17] proposed a SMC system that applies a secure summation protocol without the use of a trusted third party. The proposed protocol focuses on the increased computation complexity to avoid hacking. Each agent splits its data to a fixed number of segments and promotes a single segment to the next agent at each iteration. As an extension Sheikh et al. [17] define a master agent, which sets a random number during the initialization. Despite the fact that this protocol does not utilize a third trusted party, it is weak because if two neighbour agents collude, they can reveal the data value of the middle agent. Moreover, this technique imposes a considerable overhead in the communication between the agents.

Many recent research studies focus on privacy preserving on vertical and on horizontal partition of data. Our approach is oriented to horizontally distributed data, as each AAL agent keeps a private database with its values and each database contains the same set of attributes. Specifically, Karr et al. [8] propose a secure computation of linear regression for horizontally partitioned data without the use of a trusted third party. This is achieved by converting the linear regression equation to a summation form, where the quantities of each summation involve attribute values of the same agency. To protect data from the scope of the source and the values, they propose a SMC secure sum computation protocol. During the initialization of the protocol, a master agent adds his private value with a random number, that he previously produced, and forwards the summed value to the next agent. Each agent receives the aggregated value from the previous agent and forwards it to the neighbor agent, after the addition of his private value. The total summation result is returned back to the master agent, which removes his random number. This protocol is weak mostly because a private value of an agent can be revealed by the collusion of his neighbors. Also due to the circular mode of the algorithm, it can not be parallelised.

The study of Molina et al. [14] is closer to our approach. Specifically, they propose an application of homomorphic encryption to compute basic statistics on aggregated medical data which also guarantee the privacy of the medical data. Their SMC protocol preserves the privacy between the caregivers, where each one computes statistics for their corresponding patients. This is achieved with a double encryption, each one depending on a different public key — the public key of researcher and the public key of a caregiver chosen randomly to work as the aggregator. This approach can be mapped well in distributed systems because each caregiver can work in parallel to compute aggregates of their patients' data. However, privacy is relatively weak as the researcher and the aggregator can collude to reveal the plaintexts of each caregiver. Moreover, doubly homomorphic encryption schemes are not fully explored to define which statistics can be determined.

3 Privacy-Preserving Statistical Analysis

In this section we introduce our system architecture, and show how elementary statistical analysis methods can be implemented within this architecture in a way that essencially preserves the API of their convential implementation. As a showcase, we assume the R language implementation of the t-test and show how the same interface can be implemented within our privacy-preserving architecture instead of by directly accessing data matrices. As the architecture assumes the existence of a privacy-preseving summation protocol to access the private data, we also discuss what characteristics are required from this protocol.

3.1 System Architecture

The system architecture can be perceived as a stack of three layers and each layer depends on the functionality provided from the layer at the lower stage. The upper layer, called the *Medical Researcher's interface*, accepts from the medical researcher the method with the initial parameters to be executed by the system. The purpose of this interface is to provide a familiar environment to the researcher and therefore in our current implementation this layer is developed in the R language. The initial parameters are transformed appropriately in order to be passed to the next layer, which is the *Compilation Layer*. At that stage, the high-level parameters and commands of the statistical method are transformed into low-level instruction for accessing the private databases of the agents. An instruction represents an aggregation over a selection of data. Currently, the aggregation operation is summation. However, the aggregations that are on one hand feasible by the system and on the other hand safe for preserving privacy depend on the secure protocol used. These instructions will be eventually evaluated by the lowest layer of the architecture, the *Privacy Protocol Layer*. Figure 1 depicts the system architecture and the information exchanged between the layers.

The Medical Researcher's interface The interface is developed in the R language since it offers a variety of plotting and analysis tools, while in parallel it is a familiar environment for statisticians. The researchers can execute the statistical method through the R environment by importing the *secure statistics* library. The purpose of this library is to expose high-level statistic methods (e.g. linear regression, t-test) as R functions.

The *secure statistics* library receive the same arguments as the conventional statistics functions in R. The only difference is that the data arguments are not matrices of values, but the parameters needed to make a distributed computation. The results of the statistics functions are, then, identical to those of the respective conventional functions over the same data.

The Compilation Layer This layer is responsible for the communication between the two other layers. Specifically, it translates the arguments of the *secure statistic* to a suitable format, thus it defines the appropriate data that are going to be used for the statistic computation. Moreover, it converts the simple statistic equations to a set of summations; a compatible format to achieve the secure

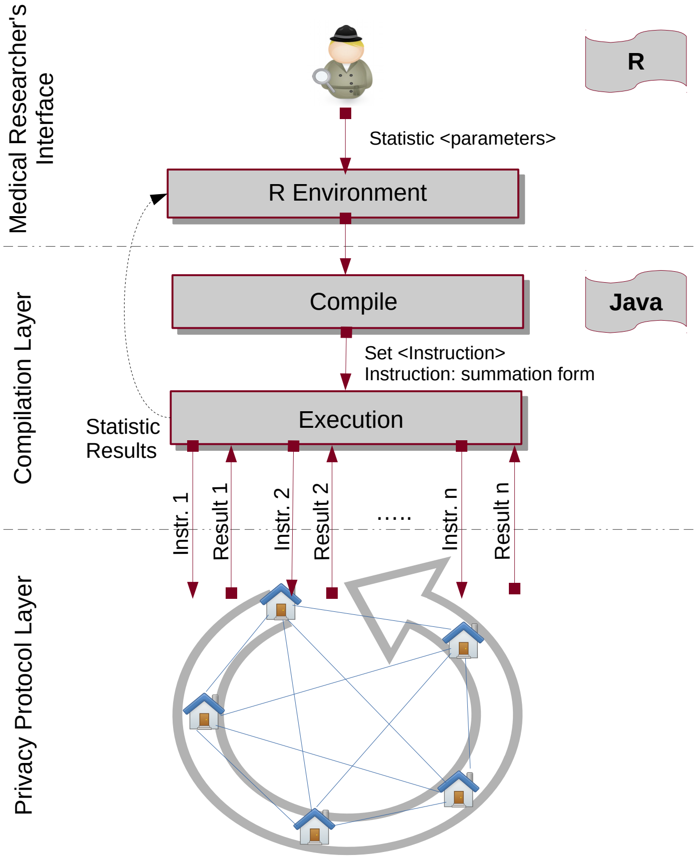

Fig. 1. The system's architecture

summation protocol. Therefore, a set of instructions is composed where each instruction represents a summation equation of the statistic with the appropriate parameters set for its computation. During the execution, the compilation layer gives to the privacy protocol layer a single instruction at a time and it receives its result. After the execution of the whole set, it computes the statistic and the analysis parameters. The statistic result is sent back to the Medical Researcher's interface layer.

The Aggregation Protocol This layer executes the privacy protocol between the AAL agents, To deal with the concurrent computation of each instruction, we model our agents as actors. Each actor makes the appropriate computations with respect to the given instruction and its private data. These computations can easily be done since every AAL agent controls its corresponding health records. After the computation of the value, which represents the initial secret, the privacy protocol is executed. The protocol may involve all the actors to work collaboratively in order to compute the aggregation of their secrets without revealing the actual secrets to each other or the agent requesting the aggregation. The aggregated result is collected a designated actor. The selection of such actor is irrelevant and can be done randomly. Our proposed implementation for this layer is presented in more detail in Section 4.

Example We will use a simple example to better demonstrate the proposed system. Suppose that a medical researcher needs to run a t-test to assess whether the means of two groups are statistically different from each other, that is to compute t in Eq. 1:

$$t = \frac{\bar{X} - \bar{Y}}{\sqrt{|X|^{-1}\,\sigma_X^2 + |Y|^{-1}\,\sigma_Y^2}} \tag{1}$$

where X and Y are the datapoints of the two groups, $\bar{X}$ and $\bar{Y}$ are their means, $|X|$ and $|Y|$ are their cardinalities, and σ_X^2 and σ_Y^2 their variances.

Assume, for instance, that a researcher wants to test the effect of medicine M_1 (Group 1) and medicine M_2 (Group 2) on blood pressure, with the further restriction that participants in Group 2 should be above 65 years old. A workflow using the R language would be:

- Select from a database the instances that match Group 1 criteria and store them in variable X
- Select from a database the instances that match Group 2 criteria and store them in variable Y
- Decide on the conditions of the T-test, such as the confidence level and alternation, and store them in variable C
- Pass X, Y, C as arguments to an implementation of t-test

Our architecture allows this workflow to remain essentially unaffected, except for the contents of X and Y. Instead of holding actual data arrays these now contain a representation of the Group 1 and Group 2 criteria, so that the selection can be executed in distributed manner. Using this representation, a privacy-aware implementation of t-test can produce the exact same result as the conventional implementation, except without ever accessing any individual data.

This representation declares a list of dependent variables and a list of eligibility criteria of the sample groups, as a set of (variable, operator, value) tuples. In our example, we assign to X and Y the criteria that we would have used to assign to them a value array if we had full access to the data:

- $X = [(\text{"medicine"}, =, \text{"M}_1\text{"})]$
- $Y = [(\text{"medicine"}, =, \text{"M}_2\text{"}),\ (\text{"age"}, >, \text{"65"})]$

The compilation layer converts the t-test implementation into a set of instructions. Recall that each instruction is an aggregation over the private data

Function	Definition
add(C)	$\sum_i s_i(C)$, where $s_i(C)$ is the secret value of the i-th AAL agent if condition C is satisfied, 0 otherwise
add$^2(C,k)$	$\sum_i (s_i(C)+k)^2$, where k is a constant and $s_i(C)$ is same as above
cnt(C)	$\sum_i c_i(C)$, where $c_i(C)$ is 1 if the i-th AAL agent satisfies condition C, 0 otherwise.

Table 1. Characteristic instructions provided by the RASSP Protocol.

of each agent, under the given selection restrictions. Table 1 defines the instructions needed to implement the t-test (Eq. 1), which is then implemented using the following pseudo-code:

1. $X = [(\texttt{"}\text{medicine}\texttt{"}, =, \texttt{"}\text{M}_1\texttt{"})];$
   X is a representation of the secret values of all AAL agents where medicine M_1 is used.
2. $Y = [(\texttt{"}\text{medicine}\texttt{"}, =, \texttt{"}\text{M}_1\texttt{"}), (\texttt{"}\text{age}\texttt{"}, >, \texttt{"}65\texttt{"})];$
   Y is a representation of the secret values of all AAL agents where medicine M_2 is used and age is above 65.
3. $N_1 = \text{add}(X); N_2 = \text{add}(Y);$
   N_1 is the sum of the secret values X and N_2 is the sum of the secret values Y.
4. $C_1 = \text{cnt}(X); C_2 = \text{cnt}(Y);$
   C_1 is the number of AAL agents with non-zero values in X and C_2 is the number of AAL agents with non-zero values in Y.
5. $\bar{X} = N_1/C_1; \bar{Y} = N_2/C_2;$
   This uses the values above to calculate means.
6. $\sigma_X^2 = \text{add}^2(X, -\bar{X}); \sigma_Y^2 = \text{add}^2(Y, -\bar{Y});$
   This uses the values above to calculate variances.
7. $T = \left(\bar{X} - \bar{Y}\right) / \text{sqroot}\left(\sigma_X^2/C_1 + \sigma_Y^2/C_2\right);$

Each instruction is executed with the use of the secure summation protocol, obtaining the aggregate values specified in the instruction without obtaining the values themselves. From the perspective of the R interface user, the t-test functions operate as if they had been passed the actual value matrices as parameters.

3.2 Reference Implementation

The system architecture that is described in Section 3.1 is implemented by the open source project at https://bitbucket.org/dataengineering/rassp

The project's source code is organized in three modules, each one implementing one of the layers in our architecture:

- `proto` implements the *aggregation protocol*

- `stats` is the implementation of statistical analysis primitives over an aggregation protocol, and implements the *compilation layer*
- `RStats` implements the R interface for the medical researcher over the compilation layer.

To execute the example immediatelly above using our implementation, the medical researcher executes the following code in the R interface:

```
# Describe the two groups in GroupStat structures:
group1 <- GroupStat(list(c("med","=","A")))
group2 <- GroupStat(list(c("med","=","B"), c("age",">","65")))
# Set dependent variables and groups in a Parameters structure:
p <- Parameters(list("bloodPr"), list(group1, group2))
# Execute the normal t-test using the Parameters structure p:
ttest(p, varEq=TRUE)
```

What is important to note in the example is that our implementation of the `ttest()` function presents an interface identical to the standard R implementation of the t-test. The underlying difference is that the `Parameters` structure does not point to actual data matrices but to instances of our `GroupStat` structure, which hold the information needed by the compilation layer in order to distibute the computation to the participating nodes.

3.3 Discussion

The proposed system architecture assumes that:

- The statistical analysis that is to be carried out can be implemented using the set of aggregation instructions provided by the aggregation protocol. In other words the algorithm should not depend on individual data points.
- A summation protocol exists that guarantees privacy.

The first assumption holds, since the most commonly used class of data mining algorithms can be expressed as an iteration of summation expressions [9]. If needed, categorical operators can be implemented based on summation [12].

Regarding the second assumption, we will now proceed to discuss the summation protocols that can be used in our architecture and, in Section 4, present the protocol we use in our reference implementation of the architecture.

Most of the related studies guarantee their privacy by utilising encryption or differential privacy techniques. These approaches do not fit in our problem, because we deal with medical history data that are distributed across AAL agents. In homomorphic techniques, a *master agent* shares a public key with the rest of the agents, in order to encrypt their data, and keeps a private key for the final decryption. Such a mechanism is privately weak in the case of collaborative computations, because if the medical researcher (master agent) and one AAL agent collude, they can learn another AAL agent's private value. This makes the technical protocol weak, as it places a heavy burden on non-technical policies and protocols to guarantee the integrity of the medical researcher. Since our main aim is to alleviate the need for non-technical policies and protocols and to make it easier for medical researchers to run statistics over datapoint they are not meant to access directly, homomorphism encryption does not cover our requirements.

In addition, differential privacy is also not applicable, from both the perspective of the medical researcher as well as from that of the AAL agent. From the perspective of the medical researcher, differential privacy computes *approximations*, which can be a problem as discussed in Section 2 above. From the perspective of the AAL agent, the secret value can be approximated by its repeated querying, since a different perturbation of the real secret needs to be computed for each query. The AAL agent cannot produce a single perturbed value and use this for all queries, since it needs to be re-computed to follow the distribution parameters requested by the medical researcher. This might be less of a problem in time-series data (such as power grid data or traffic data), but can result in substantial information leaking in static historical data, such as health records.

4 The Secure Summation Protocol

4.1 Background

Secret sharing schemes divide a secret into many *shares* which can be distributed to n mutually suspicious agents. The initial secret can be revealed if any k of these n agents combine their shares. We will call such schemes, (k, n)-threshold schemes. If such a scheme also possesses the *homomorphism* property, then multiple secrets can be combined by direct computation only on the shares. Such schemes are usually called *composite secret sharing schemes* [2].

More specifically, assume n mutually suspicious agents and each agent holds a secret s_i. The desired computation is combination into a super-secret s under an operation $\oplus$, namely $s = s_1 \oplus \cdots \oplus s_n$. Using a secret sharing scheme each s_i can be split into k shares $d_{i_1}, \ldots, d_{i_k}$ such that given a known function F_I it is the case that:

$$s_i = F_I(d_{i_1}, \ldots d_{i_k})$$

We will say the (k, n) threshold scheme has the $(\oplus, \otimes)$-homomorphism property if whenever $s = F_I(d_1, \ldots, d_k)$ and $s' = F_I(d'_1, \ldots d'_k)$ then

$$s \oplus s' = F_I(d_1 \otimes d'_1, \ldots, d_k \otimes d'_k)$$

The composition of the shares d_1, d'_1 yield a *super-share* $d_1 \otimes d'_1$. In other words, the $(\oplus, \otimes)$-homomorphism property implies that the composition of the shares under the operator $\otimes$ are shares of the composition under the operator $\oplus$.

Overall, the advantage of having a composite secret sharing scheme is that secret cannot be obtained, only if k or more agents collude and combine their sub-shares. In addition, this protocol is suitable to our approach from the AAL agent's point of view, because it does not use a trusted third party or depends on cryptographic assumptions, while at the same time it is k-secure. This approach represents a secure summation protocol that can easily be applied to collaborative agent systems.

Based on this mathematical foundation, we will now proceed to present the RASSP protocol, a $(+,+)$-homophorphic composite secret sharing scheme.

4.2 The RASSP Protocol

Assume that we have n AAL agents, where each one has its private value $v_i, i \in [1..n]$. Each AAL produces random breakdown of v_i into n terms $r_{ij}, j = 1..n$ such that $v_i = \sum_{j=1}^{n} r_{ij}$. These terms are computed by first producing $n-1$ random terms $r_{ij}, j = 1..i-1, i+1..n$ and then setting

$$r_{ii} = v_i - \sum_{j \in [1..n]-\{i\}} r_{ij}$$

The r_{ij} terms are called *sub-shares* and are (except for r_{ii}) shared with the rest of the AAL agents, one per agent. In this manner, each AAL agent shares $n-1$ values and receives $n-1$ values from the rest of the AAL agents. The *super-share* Y_i for each agent is defined as:

$$Y_i = r_{ii} + \sum_{k \in [1..n]-\{i\}} r_{ki} \tag{2}$$

Notice how the super-share of AAL agent i is the sum of the sub-share that it has not shared and of all the sub-shares that it has received from the other AAL agents. Finally, we define a function F_I as the sum of the super-shares:

$$F_I(Y_1, \ldots, Y_n) = \sum_{i=1}^{n} Y_i \tag{3}$$

It is straightforward to verify that $F_I(Y_1, \ldots, Y_n)$ is equal to the sum of all secrets. It is also straightforward to verify that only random numbers and obscured data values are shared between AAL agents and between AAL agents the researcher. Notice also that only if $(n-1)$ AAL agents collude to merge their sub-shares can the private value of the n-th agent be revealed. Therefore, our system guarantees $(n-1)$-security.

Figure 2 gives an example of the above description for a system of three AAL agents. In this example House1 has the private value v_1 and produces three numbers: r_{11}, r_{12}, r_{13}. Then, it shares r_{12} and r_{13} with House2 and House3, keeping r_{11} hidden. House1 receives two numbers (r_{21}, r_{31}) from the other AAL agents. In then shares the computed Y_1, so that F_I can be computed by summing all Y_i. $F_I(Y_1, Y_2, Y_3)$ computes the sum of all three AAL agents' secret values.

The described secure summation protocol is suitable for computing medical statistics and preserve privacy at the same time. The only constraint is that

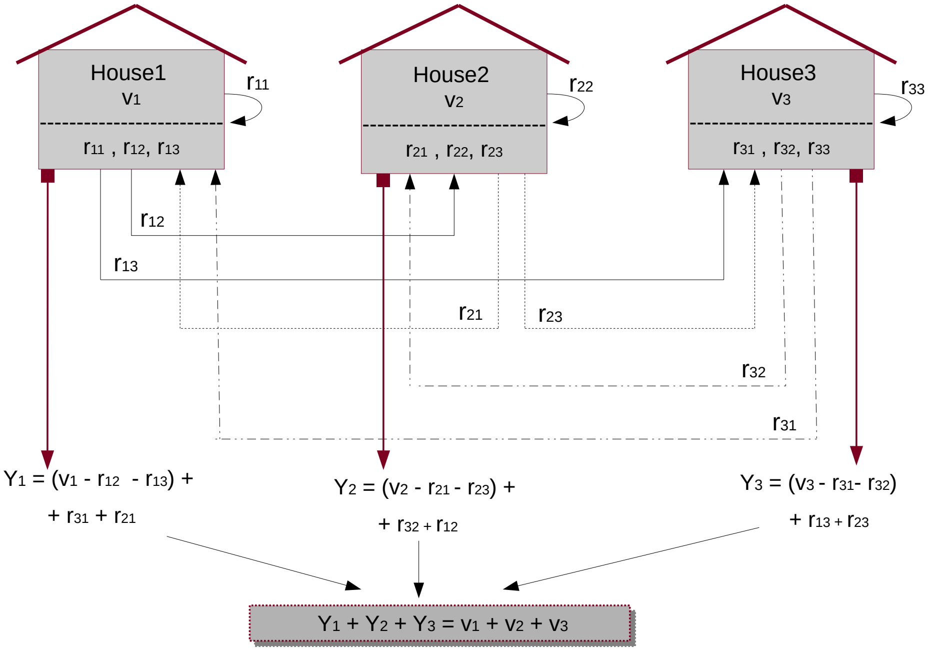

Fig. 2. The RASSP secure summation protocol.

the resulted outcome is a sum of the private values, thus the statistic equations should be converted in a summation form. The summation form results in accurate values and not approximations, while simultaneously it can easily be parallelised [3]. Besides, medical researchers typically use descriptive statistics which utilise numerical descriptors such as mean and standard deviation. These descriptors can easily be converted into a summation form, thus they can be computed by our system.

5 Conclusions and Further Work

With this paper we experiment with privacy-preserving data mining that is accessed not through specialized APIs and tools, but through statistical analysis tools that are ubiquitous in data-driven research, such as the R language and its statistical analysis libraries. In this manner, we are targeting the uptake of our privacy-preservation infrastructure by the medical research community, as the discussion around privacy preservation is mute if the data cannot be efficiently and effectively used to achieve the medical research purpose.

Specifically, our first contribution is our architecture and its reference implementation. This architecture foresees the primitive *instructions* needed to re-implement the elementary statistical methods so that they only access data via a privacy-preserving protocol. The overall gist is that function arguments remain the same, except for substituting data matrices with a specification of how to select the data that each AAL agent will contribute to the distributed computation. The advantage is that more complex analysis and visualisation tools that are built upon these elementary methods can remain unaffected by replacing direct access to data with access via privacy-preserving protocols.

A further contribution is our review of *secure multiparty computation*, *differential privacy*, and *homomorphic encryption* approaches to justify and explain assuming the former as the most appropriate basis for our personal health record use case. Finally, we introduce RASSP, a secure summation protocol that computes sums within Benaloh's *composite secret sharing* framework for secure multiparty computation.

More complex instructions will be implemented as iterations of the primitive sum operator. As this is bound to reduce the run-time efficiency of the system, our next steps will be to integrate distributed computation concepts in order to parallelise the computation. Chu et al. [3] propose using the map-reduce framework to execute a variety of statistics, where the summation form of their equations facilitates the distribution of their calculations. This approach will be mapped to our architecture to improve the run-time efficiency of the system. A further optimization step will be to execute simultaneously instructions when there is no dependence between them. To achieve this, we will transfer from the programming languages and distributed computation literature optimization methods that can decide about the most efficient execution plan for a given program with multiple calls to the primitive instructions. We will also need to extend the current API of these instructions in order to allow multiple requests to be made to the distributed AAL agents with one network transaction.

Another relevent on-going discussion in the community is the involvement of humans in the data mining process itself [6] and when acting upon data mining results [11]. Given the responsibility of the medical practitioner when using data to make medical decisions, the uptake of our—and, in fact, any—privacy preserving protocol depends on the data consumers' ability to apply checks and bounds to the values that are allowed to participate in the computation. In this context, a more ambitious goal is to extend the RASSP protocol so that the medical researcher can specify what value ranges of the secret variables are reasonable or useful and to have this range restriction guaranteed without having to trust the AAL agents. To achieve this, we will experiment with variations of the current RASSP protocol where the AAL agents can to some extend also check whether their peers are (erroneously or maliciously) sharing sub-shares that add up to out-of-range values. Since such AAL agent behaviour can corrupt the result or obscure useful outliers, it undermines the trust that the researcher places on the result. Our goal is to devise a system of cross-checks that makes it unlikely that out-of-range values contribute to the sum undetected, but without compromising the privacy-preserving nature of the protocol.

Acknowledgements

The work described here was carried out in the context of the RADIO project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 643892. For more details, please visit the RADIO Web site http://www.radio-project.eu

References

[1] Ajmani, S., Morris, R., Liskov, B.: A trusted third-party computation service. Tech. rep., MIT-LCS-TR-847, MIT (2001)

[2] Benaloh, J.C.: Secret sharing homomorphisms: Keeping shares of a secret secret. In: Advances in Cryptology: Proceedings of CRYPTO '86. LNCS, vol. 263, pp. 251–260. Springer (1986)

[3] Chu, C.t., Kim, S.K., Lin, Y.A., Yu, Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19: Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS 2007), Vancouver, BC, Canada, 3–5 December 2007. pp. 281–288. MIT Press (2007)

[4] Clifton, C., Kantarcioglu, M., Vaidya, J.: Defining privacy for data mining. In: Proceedings of the National Science Foundation Workshop on Next Generation Data Mining, Baltimore, USA, 1-3 November 2002 (2002)

[5] Hanmanthu, B., Ram, B.R., Niranjan, P.: Third party privacy preserving protocol for perturbation based classification of vertically fragmented data bases. arXiv preprint arXiv:1304.6575 (2013)

[6] Holzinger, A.: Interactive machine learning for health informatics: When do we need the human-in-the-loop? Brain Informatics 3(2), 119–131 (2016)

[7] Horvitz, E., Mulligan, D.: Data, privacy, and the greater good. Science Magazine (2015)

[8] Karr, A.F., Lin, X., Sanil, A.P., Reiter, J.P.: Secure regression on distributed databases. Journal of Computational and Graphical Statistics 14(2), 263–279 (2005)

[9] Kearns, M.: Efficient noise-tolerant learning from statistical queries. Journal of the ACM (JACM) 45(6), 983–1006 (1998)

[10] Kerschbaum, F.: Privacy-preserving computation. In: Privacy Technologies and Policy: Revised Selected Papers from the First Annual Privacy Forum (APF 2012), Limassol, Cyprus, 10-11 October 2012, pp. 41–54. Springer (2012)

[11] Kieseberg, P., Malle, B., Frühwirt, P., Weippl, E., Holzinger, A.: A tamper-proof audit and control system for the doctor in the loop. Brain Informatics (2016)

[12] Kissner, L., Song, D.: Privacy-preserving set operations. In: Advances in Cryptology: Proceedings of the 25th Annual International Cryptology Conference (CRYPTO 2005), Santa Barbara, California, USA, 14–18 August 2005. LNCS, vol. 3621, pp. 241–257. Springer (2005)

[13] McSherry, F.D.: Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data (SIGMOD 2009). pp. 19–30. ACM (2009)

[14] Molina, A.D., Salajegheh, M., Fu, K.: Hiccups: health information collaborative collection using privacy and security. In: Proceedings of the first ACM workshop on Security and privacy in medical and home-care systems (SPIMACS 2009). pp. 21–30. ACM (2009)

[15] Ohm, P.: Broken promises of privacy: Responding to the surprising failure of anonymization. UCLA Law Review 57 (2010)

[16] Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '99), Prague, Czech Republic, May 2-6, 1999. LNCS, vol. 1592, pp. 223–238. Springer (1999)

[17] Sheikh, R., Kumar, B., Mishra, D.K.: Privacy preserving k secure sum protocol. arXiv preprint arXiv:0912.0956 (2009)

[18] Shi, E., Chan, T.H., Rieffel, E., Chow, R., Song, D.: Privacy-preserving aggregation of time-series data. In: Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS 2011). vol. 2, pp. 1–17 (2011)