Privacy-enhancing data aggregation and data analytics in wireless networks for a large class of distributed queries

Wireless Networks
https://doi.org/10.1007/s11276-022-03108-4

Privacy-enhancing data aggregation and data analytics in wireless networks for a large class of distributed queries

Xuechao Yang[1] · Andrei Kelarev[1] · Xun Yi[1]

Accepted: 18 August 2022

Abstract
Privacy-enhancing techniques and protocols for data aggregation and analytics in wireless networks require the development of novel methods for efficient and privacy-preserving computation of distributed queries with the protection of outcomes from active attackers. Previous approaches to secure privacy-preserving computation of distributed queries incur significant communication overhead and cannot be applied to big data. This paper proposes two solutions to the problem of efficient and privacy-preserving computation of distributed queries with the protection of outcomes from active outsider attackers for a new large class of distributed statistical or numerical queries. This class contains many useful statistics and is larger than other classes considered in the literature previously. We propose two protocols for the Protection of data from Active Attackers (PAA) in the case of distributed privacy-preserving computation: PAA applying Shamir's Secret Sharing (PAA-SSS) and PAA applying homomorphic encryption (PAA-HE). The PAA-HE protocol combines the use of ElGamal and Paillier encryption schemes in one system. Theoretical analysis and experimental results show that both protocols outperform alternative approaches. PAA-HE provides stronger protection and is more efficient than PAA-SSS.

Keywords Privacy-enhancing data aggregation · Data analytics · Distributed queries · Homomorphic encryption

1 Introduction

Privacy-enhancing techniques and protocols for data aggregation and analytics in wireless networks require novel methods for efficient and privacy-preserving computation of distributed queries with the protection of outcomes from active attackers. Research on this topic belongs to the general area of distributed privacy-preserving data mining.

Previously, efficient specialised algorithms for distributed data mining and machine learning have been developed, for example, in [1–4]. On the other hand, the protection of privacy is also crucial for successful applications of novel cyber technologies [5–7]. For example, privacy-preserving techniques have been investigated for data publishing [8], service selection [9], and trend surface analysis [10].

In this context, the present paper investigates the general situation where it is necessary to ensure privacy and at the same time provide answers to statistical or numerical aggregate queries over a large distributed dataset, which is a union of several separate subsets such that the managers of the subsets are not allowed to disclose the content of their data or transfer their data to other entities or competing organisations operating in the same wireless network.

Denote by $\mathscr{D}$ the whole distributed dataset, which is a union of the subsets $\mathscr{D}_1, \ldots, \mathscr{D}_M$ supervised by different managers $\mathscr{M}_1, \ldots, \mathscr{M}_M$, where M is the number of the subsets. We assume that the dataset is horizontally distributed, i.e., each record belongs to one of the subsets $\mathscr{D}_1, \ldots, \mathscr{D}_M$. For a positive integer n, we denote by $[1 : n]$ the set $\{1, 2, \ldots, n\}$.

A client submits a query to the system being organised by the managers. The managers process the query

✉ Xuechao Yang
xuechao.yang@rmit.edu.au

Andrei Kelarev
andrei.kelarev@rmit.edu.au

Xun Yi
xun.yi@rmit.edu.au

[1] School of Computing Technology, RMIT University, 124 La Trobe St, Melbourne, VIC 3000, Australia

Published online: 14 September 2022

following the protocol and return the final outcome to the client. The client does not have access to the records of the dataset, because they contain confidential information. For $m \in [1:M]$, each manager $\mathscr{M}_m$ has access only to the records of the corresponding dataset $\mathscr{D}_m$, but has no right to access the datasets of the other managers.

The dataset managers processing distributed queries represent official established entities that own their parts of the data contained in the distributed dataset. As a typical important example, the managers may official representatives of different organisation participating in the wireless network. Therefore, it is natural to assume that the individual dataset managers are honest. Nevertheless, they cannot share confidential information information of individual entries in their part of the dataset with the representatives of other competing organisations.

An efficient noise addition framework for privacy preserving data mining was proposed in [11–14]. An algorithm for the private processing of distributed queries was proposed in [15]. The present paper considers a larger class of queries. The algorithm proposed in [15] was the first and only algorithm applicable in the situation considered in the present paper. However, the procedure proposed in [15] does not provide protection against active attackers and the previous algorithm cannot solve the problem considered in the present paper. The readers are referred to [15] for additional examples of other previous related publications.

We propose solutions to this problem by developing new protocols, which employ methods different from those used in [15]. Besides, the present paper handles a larger class of numerical queries in comparison with [15]. In particular, vector functions considered in this paper are more general than the scalar functions used in [15], and our new class of functions treated by our protocols in the present paper is larger than the one considered in [15].

In our model, the managers use separate servers for secure computation. It is likely that every participant would prefer to be equally involved in the process of secure computation. There are no reasons to introduce a single trusted authority handling one server for all secure computations. Instead, we assume that each manager introduces an individual server, where all managers have to communicate intermediate values in order to organize the process. Accordingly, we assume that the managers $\mathscr{D}_1, \ldots, \mathscr{D}_M$ introduce the servers $S_1, \ldots, S_M$, where the server S_m belongs to $\mathscr{M}_m$, for $m \in [1:M]$.

However, the problem of protecting the outcomes of distributed queries from active attackers has not been considered in this setting. This problem is important because the servers are likely to be targeted by active outsider attackers as they are new and each of them is involved in communication with the managers of all subsets and contains a lot of confidential information contributed by all the individual managers. This is why the problem of protecting the outcomes of distributed queries from active outsider attackers is paramount. This problem has not been considered before.

When the active attackers compromise several of the servers $S_1, \ldots, S_M$, this leads to occurrences of Byzantine faults in the compromised servers. Let k be a positive integer, $k < M$, equal to the largest number of the servers $S_1, \ldots, S_M$ which can be compromised by active outsider attackers. This integer is an input parameter to our protocols.

The aim of this paper is to propose solutions to the problem of protecting the outcomes of distributed queries from active outsider attackers. We define a new large class of distributed queries to be handled by our protocols. A formal definition of this class is given in Sect. 3. This class contains many statistical queries of practical value. We propose solutions to the problem of protecting all queries from this class against active outsider attackers. We introduce two recursive protocols for the Protection against Active Attackers (PAA). The two variants of our recursive PAA protocols are the PAA, applying Shamir's Secret Sharing (PAA-SSS), and PAA, applying homomorphic encryption (PAA-HE). The latter combines the ElGamal and Paillier encryption schemes in order to handle certain steps of the whole system. Theoretical analysis and the results of our experiments show that (i) both protocols significantly outperform different more straightforward approaches, and (ii) PAA-HE provides stronger protection to the query outcomes, but is slower than PAA-SSS.

The paper comprises the following sections. Section 3 explains the PAA-SSS and PAA-HE protocols. It introduces the class $\mathscr{C}$ of queries handled by the protocols, explains iterations and steps of the PAA-SSS and PAA-HE protocols, and shows that the class $\mathscr{C}$ contains many important statistical queries. The experiments comparing PAA-SSS and PAA-HE with other algorithms are presented in Sect. 4. Section 5 concludes the paper.

2 Preliminaries

2.1 ElGamal encryption scheme for $\mathscr{M}_1, \ldots, \mathscr{M}_M$

Following [16, Sect. 2.3], here we define succinct notation for the ElGamal encryption scheme introduced in [17]. Each manager $\mathscr{M}_\ell$, $\ell \in [1:M]$, chooses a secret key sk_ℓ

and a public key pk_ℓ, as explained in [16, Sect. 2.3]. Let us denote by $c = \mathsf{E}_{EG}(t, \mathsf{pk})$ the ElGamal encryption of a plaintext t. We denote by $t = \mathsf{D}_{EG}(c, \mathsf{sk}_\ell))$ the ElGamal decryption of c.

For any plaintexts $t_1, \ldots, t_M$, the ElGamal cryptosystem satisfies the following homomorphic property:

$$\prod_{m=1}^{M} \mathsf{E}_{EG}(t_m, \mathsf{pk}_\ell) = \mathsf{E}_{EG}\left(\prod_{m=1}^{M} t_m, \mathsf{pk}_\ell\right). \tag{1}$$

For more explanations and examples, the readers are referred to [16, Sect. 2.3].

2.2 Paillier encryption scheme for $\mathscr{M}_1, \ldots, \mathscr{M}_M$

Following [16, Sect. 2.4], we introduce concise notation for the Paillier encryption scheme introduced in [18]. Each manager $\mathscr{M}_\ell$, $\ell \in [1:M]$, chooses a secret key sk'_ℓ and a public key pk'_ℓ, as explained in [16, Sect. 2.4]. We denote by $c = E_P(t, \mathsf{pk}'_\ell)$ the Paillier encryption of the plaintext t. Let us denote by $D_P(c, \mathsf{sk}'_\ell)$ the Paillier decryption of c.

For any plaintexts $t_1, \ldots, t_M$, the Paillier encryption scheme satisfies the following homomorphic property:

$$\prod_{m=1}^{M} \mathsf{E}_P(t_m, \mathsf{pk}'_\ell) = \mathsf{E}_P\left(\prod_{m=1}^{M} t_m, \mathsf{pk}'_\ell\right). \tag{2}$$

For more details and examples, the readers are referred to [16, Sect. 2.4].

2.3 Shamir's secret sharing for $\mathscr{M}_1, \ldots, \mathscr{M}_M$

To apply Shamir's secret sharing [19] as explained in [20], the managers choose a finite field $\mathscr{F}$ with $\|\mathscr{F}\| > M$ and with a primitive M-th root of unity, $\alpha \in \mathscr{F}$, $\alpha^M = 1$. All values of the data are represented as elements of $\mathscr{F}$. They compute $\xi_1 = \alpha^0$, $\xi_2 = \alpha^1, \ldots, \xi_M = \alpha^{M-1}$ in $\mathscr{F}$.

Suppose that each manager $\mathscr{M}_m$, $m \in [1:M]$, has a secret value y_m. To introduce it to the process, Shamir's secret sharing [19] is used as follows. Recall that the smallest integer that is greater than or equal to x is denoted by $\lceil x \rceil$. Putting $k = \lceil M/2 \rceil - 1$, the $\mathscr{M}_m$ selects k random elements $u_{m,1}, \ldots, u_{m,k} \in \mathscr{F}$, defines the polynomial $g_m(x) = y_m + u_{m,1}x + \cdots + u_{m,k}x^k$, and for all $m' \in [1:M]$ sends each value $\Omega_{m'}(y_m) = g_m(\xi_{m'})$ as a secret share to $\mathscr{M}_{m'}$. The secret value y_m has been split into secret shares

$$\Omega_1(y_m) = g_m(\xi_1), \ldots, \Omega_M(y_m) = g_m(\xi_M). \tag{3}$$

These shares encode y_m, because Lagrange's interpolation formula

$$g_m(x) = \sum_{m'=1}^{k} \left(\Omega_{m'}(y_m) \frac{\prod_{m'' \in [1:k], m'' \neq m'} x - \xi_{m''}}{\prod_{m'' \in [1:k], m'' \neq m'} \xi_{m'} - \xi_{m''}} \right) \tag{4}$$

restores the polynomial $g_m(x)$ from (3) and recovers $y_m = g_m(0)$.

It is explained in [20] how each manager $\mathscr{M}_m$, $m \in [1:M]$, can privately compute two new values $\Theta_m(y_1, \ldots, y_M)$ and $\Delta_m(y_1, \ldots, y_M)$, which encode the sum $\sum_{m'=1}^{M} y_{m'}$ and the product $\prod_{m'=1}^{M} y_{m'}$ as their corresponding secret shares, respectively. In order to refer to these values, we denote them by

$$\Theta_m(y_1, \ldots, y_M) = \Omega_m\left(\sum_{m'=1}^{M} y_{m'}\right), \tag{5}$$

$$\Delta_m(y_1, \ldots, y_M) = \Omega_m\left(\prod_{m'=1}^{M} y_{m'}\right). \tag{6}$$

3 The PAA protocols

We consider the general case in which the client communicates to the managers of the distributed dataset $\mathscr{D}$ a query of the form (φ, B), where B is a finite set of Boolean expressions indicated by the client for choosing vectors to be included in the query, and φ is a vector function selected by the client for computing the query outcome. The main notations used in our protocols are listed in Table 1.

The queries handled by the PAA-SSS and PAA-HE protocols incorporate Boolean expressions B. They are

Table 1 Main notation used in this paper

Number of independent managers	M
Individual managers	$\mathscr{M}_1, \mathscr{M}_2, \ldots, \mathscr{M}_M$
Their separate datasets	$\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_M$
Combined dataset	$\mathscr{D} = \mathscr{D}_1 \cup \cdots \cup \mathscr{D}_M$
Dimension of vectors in dataset	C
A vector in $\mathscr{D}$	$\mathbf{v}_r = (v_{r,1}, v_{r,2}, \ldots, v_{r,C})$
Number of vectors in T and T_m	$\|T\| = R$, $\|T_m\| = R_m$
Number of vectors in $\cup_{m'=1}^{m-1} \mathscr{D}_{m'}$	$\gamma_m = \sum_{m'=1}^{m-1} R_{m'}$
Query	(φ, B) defined by (7)
Query function	φ defined by (7)
Class of numerical queries	$\mathscr{C}$
Servers	$S_1, S_2, \ldots, S_M$
Number of compromised servers	k

indicated by the client and used to select subsets of records or vectors in $\mathscr{D}$. Each of these expressions can be applied to any vector $\mathbf{v}$ in $\mathscr{D}$ and produces TRUE or FALSE for each vector.

Denote by $\mathscr{B}$ the class of all Boolean expressions considered in this paper. The class $\mathscr{B}$ is defined recursively as follows. First, $\mathscr{B}$ contains all Boolean basic expressions of the form $\psi(\mathbf{v}) = \varphi(\mathbf{v})$, $\psi(\mathbf{v}) < \varphi(\mathbf{v})$, $\psi(\mathbf{v}) \leq \varphi(\mathbf{v})$, $\psi(\mathbf{v}) > \varphi(\mathbf{v})$, $\psi(\mathbf{v}) \geq \varphi(\mathbf{v})$, where ψ and φ are any numerical functions defined for any vector $\mathbf{v}$ in $\mathscr{D}$. Second, if $B_1, B_2 \in \mathscr{B}$, then the following expressions also belong to $\mathscr{B}$: $\neg B_1$, $B_1 \wedge B_2$, $B_1 \vee B_2$, $B_1 \mid B_2$, $B_1 \rightarrow B_2$, $B_1 \leftrightarrow B_2$ where $\neg$ (NOT), $\wedge$ (AND), $\vee$ (OR), $\mid$ (XOR), $\rightarrow$ (implication), and $\leftrightarrow$ (equivalence) are the well-known Boolean operators. These two rules recursively define all Boolean expressions in the class $\mathscr{B}$.

Given a finite set $B \subseteq \mathscr{B}$ specified by the client, let us denote by $T = B(\mathscr{D})$ the set of records or vectors selected in the dataset $\mathscr{D}$ by the set B of Boolean expressions. The set T consists of all vectors $\mathbf{v} \in \mathscr{D}$ such that $B(\mathbf{v}) =$ TRUE for all $B \in \mathscr{B}$.

It is easy for the managers to apply the Boolean expressions in the finite set B to their separate datasets, because it follows from the recursive definition given above that every Boolean expression applies to each vector of the dataset considered individually in isolation from other vectors. Therefore, every manager $\mathscr{M}_m$, $m \in [1:M]$, can select all vectors of the corresponding subset $\mathscr{D}_m$ locally without consulting any other manager. For $m \in [1:M]$, denote by $T_m = B(\mathscr{D}_m)$ the subset consisting of all vectors in $\mathscr{D}_m$ satisfying all Boolean expressions in the finite set B. The set T_m consists of all vectors $\mathbf{v} \in \mathscr{D}_m$ such that $B_1(\mathbf{v}) =$ TRUE for all $B_1 \in B$. Let $R = |T|$ be the cardinality of the set T, and let $R_m = |T_m|$ be the number of vectors in T_m. Then $R = \sum_{m=1}^{M} R_m$ and $T = T_1 \dot{\cup} T_2 \dot{\cup} \ldots \dot{\cup} T_M$ is a disjoint union of the sets $T_1, T_2, \ldots, T_M$.

Denote all vectors in the set $T = B(\mathscr{D})$ by $\mathbf{v}_1, \ldots, \mathbf{v}_R$. Let C be the number of components or coordinates in every vector $\mathbf{v}$ of the whole dataset $\mathscr{D}$. Denote the components of the vector $\mathbf{v} \in \mathscr{D}$ by $v_1, v_2, \ldots, v_C$. This means that $\mathbf{v} = (v_1, v_2, \ldots, v_C) \in \mathscr{D}$. For $r \in [1:R]$ and $\mathbf{v}_r \in T$, denote the components of the vector $\mathbf{v}_r$ by $v_{r,1}, \ldots, v_{r,C}$. Then we have $\mathbf{v}_r = (v_{r,1}, v_{r,2}, \ldots, v_{r,C}) \in T$.

Intuitively, the class $\mathscr{C}$ of queries handled by our protocols consists of all pairs (φ, B), where B is a finite set of Boolean expressions, and where φ is any vector function, which applies to the set $T = B(\mathscr{D})$ and which can be defined by using some vector functions of individual vectors in T, the symbols of sum $\sum_{r=1}^{R}$ and product $\prod_{r=1}^{R}$, as well as a compound vector function combining them.

More formally, the class $\mathscr{C}$ is defined as the set of all pairs (φ, B), where B is a Boolean expression and φ is a vector function defined as follows. Take any positive integers L and d. For $\ell \in [1:L]$, let d_ℓ, e_ℓ be positive integers. Let φ_1 be a vector function from $\mathbb{R}^C$ to $\mathbb{R}^{d_1}$. Let ψ_1 be a vector function from $\mathbb{R}^C$ to $\mathbb{R}^{e_1}$. Finally, let ψ be a function from $\mathbb{R}^{d_L + e_L}$ to $\mathbb{R}^d$. The function φ is defined recursively by the following equalities

$$\varphi(T) = \psi\left(\sum_{r=1}^{R} \varphi_\ell, \prod_{r=1}^{R} \psi_\ell\right), \tag{7}$$

$$\varphi_\ell = \varphi_\ell\left(\mathbf{v}_r, \sum_{r=1}^{R} \varphi_{\ell-1}, \prod_{r=1}^{R} \psi_{\ell-1}\right), \tag{8}$$

$$\psi_\ell = \psi_\ell\left(\mathbf{v}_r, \sum_{r=1}^{R} \varphi_{\ell-1}, \prod_{r=1}^{R} \psi_{\ell-1}\right), \tag{9}$$

where, for $\ell \in [2, L]$, φ_ℓ is a function from $\mathbb{R}^{C + d_{\ell-1} + e_{\ell-1}}$ to $\mathbb{R}^{d_\ell}$, and ψ_ℓ is a function from $\mathbb{R}^{C + d_{\ell-1} + e_{\ell-1}}$ to $\mathbb{R}^{e_\ell}$.

The class $\mathscr{C}$ contains many useful and well-known numerical queries. Indeed, the following smaller class $\mathscr{K}$ was defined in [15]. It consists of all functions $\varphi(T)$ given by

$$\varphi(T) = g\left(\sum_{r=1}^{R} g_1(\mathbf{v}_r), \ldots, \sum_{r=1}^{R} g_{d_1}(\mathbf{v}_r)\right), \tag{10}$$

where g is a function with d_1 arguments and $g_1, \ldots, g_{d_1}$ are scalar functions with C arguments each. If we put $L = 1, \psi = g$ and $\psi_1(\mathbf{v}_r) = (g_1(\mathbf{v}_r), \ldots, g_{d_1}(\mathbf{v}_r))$, then we get $\varphi(T) = \psi\left(\sum_{r=1}^{R} \psi_1(\mathbf{v}_r)\right)$, which is a special case of (7). Therefore $\mathscr{K} \subseteq \mathscr{C}$. It was explained in [15] that $\mathscr{K}$ contains the mean, variance, standard deviation, the coefficient of variation, the sample covariance, and the Pearson product-moment correlation coefficient (cf. [21]). It follows that all these functions also belong to $\mathscr{C}$.

Another example of a query in $\mathscr{C}$ is given by any pair (φ_1, B), where φ_1 is the *geometric mean* defined by

$$GM(v_{1,1}, \ldots, v_{R,1}) = \sqrt[R]{\prod_{r=1}^{R} v_{r,1}}. \tag{11}$$

It belongs to $\mathscr{C}$, since it is determined by (7) with $\ell = 1$, $\varphi_1(\mathbf{v}_r) = 1$, $\psi_1(\mathbf{v}_r) = v_{r,1}$, $\psi(x, y) = \sqrt[x]{y}$. It is obvious, that

$GM(v_{1,1},\ldots,v_{R,1})$ does not belong to $\mathscr{K}$. Therefore, $\mathscr{K}$ is strictly included in $\mathscr{C}$.

3.1 The PAA-SSS protocol

All steps of the PAA-SSS protocol are formally described in Algorithm 1. Let us introduce concise auxiliary notation used in Algorithm 1.

Algorithm 1 Description of the PAA-SSS protocol.

Input: The client submits (φ, B) to $\mathscr{M}_1,\ldots,\mathscr{M}_M$, where $\varphi \in \mathscr{C}$ is defined by (7).
Output: $\mathscr{M}_1,\ldots,\mathscr{M}_M$ return $\varphi(B(\mathscr{D}))$ to the client.

1: The managers choose a finite field $\mathscr{F}$ with cardinality $> M$, a primitive M-th root of unity $\alpha \in \mathscr{F}$, and compute $\xi_1 = \alpha^0, \xi_2 = \alpha^1, \ldots, \xi_M = \alpha^{M-1}$ and $A = B^{-1}PB$.
2: Set $w_0 = w'_0 = 0$.
3: **for all** $\ell = 1,\ldots,L$ **do**
4: For $m \in [1:M]$, $\mathscr{M}_m$ computes $y_m = \sum_{r=\gamma_m+1}^{\gamma_{m+1}} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$, chooses random $u_{m,1},\ldots,u_{m,k} \in \mathscr{F}$, defines $g_m(x) = y_m + u_{m,1}x + \cdots + u_{m,k}x^k$, and for all $m' \in [1:M]$ sends $z_{m,m'} = g_m(\xi_{m'})$ to $S_{m'}$.
5: As in Section 2.3, the servers compute $t_{\ell,m} = \Theta_m(z_{1,m},\ldots,z_{M,m})$, for $m \in [1:M]$, and send $t_{\ell,m}$ to all $\mathscr{M}_{m'}$, for $m' \in [1:M]$.
6: For $m \in [1:M]$, each $\mathscr{M}_m$ uses (4) to recover $w_\ell = \sum_{r=1}^{R} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ from $t_{\ell,1},\ldots,t_{\ell,M}$.
7: For $m \in [1:M]$, $\mathscr{M}_m$ computes $y'_m = \prod_{r=\gamma_m+1}^{\gamma_{m+1}} \psi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$, chooses random $u'_{m,1},\ldots,u'_{m,k} \in \mathscr{F}$, defines $g'_m(x) = y'_m + u'_{m,1}x + \cdots + u'_{m,k}x^k$, and for all $m' \in [1:M]$ sends $z'_{m,m'} = g'_m(\xi_{m'})$ to $S_{m'}$.
8: As in Section 2.3, the servers compute $t'_{\ell,m} = \Delta_m(z'_{1,m},\ldots,z'_{M,m})$, for $m \in [1:M]$, and send $t'_{\ell,m}$ to all $\mathscr{M}_{m'}$, for $m' \in [1:M]$.
9: For $m \in [1:M]$, each $\mathscr{M}_m$ uses (4) to recover $w'_\ell = \prod_{r=1}^{R} \psi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ from $t'_{\ell,1},\ldots,t'_{\ell,M}$.
10: **end for**
11: Every manager computes $\psi(w_L, w'_L) = \varphi(B(\mathscr{D}))$ and sends it to the client.

For $m \in [1:M+1]$, put $\gamma_m = \sum_{m'=1}^{m-1} R_{m'}$. In particular, $\gamma_1 = 0$. Without loss of generality, we may assume that all vectors of T are indexed so that the vectors of T_1 are indexed first and occur in succession one after another, then the vectors of T_2 follow, and so on. It follows that we can denote all vectors of T_m by $\mathbf{v}_{\gamma_m+1},\ldots,\mathbf{v}_{\gamma_m+R_m}$. Then we get

$$T_m = \{\mathbf{v}_{\gamma_m+1}, \mathbf{v}_{\gamma_m+2}, \ldots, \mathbf{v}_{\gamma_m+R_m}\}. \tag{12}$$

To be able to treat φ_1, ψ_1 in the same way as the other functions in the iterations of Algorithms 1 and 2, let us define $\varphi_1(\mathbf{v}_r, 0, 0) = \varphi_1(\mathbf{v}_r)$ and $\psi_1(\mathbf{v}_r, 0, 0) = \psi_1(\mathbf{v}_r)$. In the beginning of Algorithm 1, the managers set up Shamir's Secret Sharing scheme as indicated in line 1 and as explained in Sect. 2.3. Line 2 of Algorithm 1 sets up initial values required to start iterations of the loop in lines 3 to 10. Each iteration of the loop assumes that, for $\ell \in [1:L]$, the values $w_{\ell-1}$ and $w'_{\ell-1}$ have already been determined in the previous iteration or in line 2. Each manager locally computes the auxiliary subsum y_m in line 4 and uses Shamir's Secret Sharing to send its private share $z_{m,m'}$ to the server $S_{m'}$, for all $m' \in [1:M]$. In line 5 of Algorithm 1, the $t_{\ell,1},\ldots,t_{\ell,M}$ encode $w_\ell = \sum_{r=1}^{R} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ in (7), as explained in Sect. 2.3. As indicated in line 6, each manager $\mathscr{M}_m$ can determine the sum w_ℓ, which is a part of (7). In line 7, each manager locally computes the subproduct y'_m and sends its private shares $z'_{m,m'}$ to the servers $S_{m'}$, for all $m' \in [1:M]$. In line 8 of Algorithm 1, the $t'_{\ell,1},\ldots,t'_{\ell,M}$ encode $w'_\ell = \prod_{r=1}^{R} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ in (7), as explained in Sect. 2.3. In line 6, each manager $\mathscr{M}_m$ recovers the product w'_ℓ, which is a part of (7), as indicated in line 9. It follows from (7) that $\psi(w_L, w'_L) = \varphi(B(\mathscr{D}))$. The managers compute $\psi(w_L, w'_L)$ locally and send it to the client in line 11 of Algorithms 1.

Algorithm 2 Description of the PAA-HE protocol.

Input: The client submits (φ, B) to $\mathscr{M}_1, \ldots, \mathscr{M}_M$, where $\varphi \in \mathscr{C}$ is defined by (7).
Output: $\mathscr{M}_1, \ldots, \mathscr{M}_M$ return $\varphi(B(\mathscr{D}))$ to the client.

1: Each manager sets up their own ElGamal and Paillier encryption schemes as in Sections 2.1, 2.2.
2: For $m \in [1:M]$, $\mathscr{M}_m$ sends public Paillier key pk_m and public ElGamal key pk'_m to all other managers.
3: Set $w_0 = w'_0 = 0$.
4: **for all** $\ell = 1, \ldots, L$ **do**
5: For $m \in [1:M]$, $\mathscr{M}_m$ computes $y_m = \sum_{r=\gamma_m+1}^{\gamma_{m+1}} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$.
6: For $m, m' \in [1:M]$, each $\mathscr{M}_m$ computes $e_{m,m'} = E_P(y_m, \mathsf{pk}_{m'})$ and sends it to the server $S_{m'}$.
7: For $m \in [1:M]$, the server S_m computes $p_m = \prod_{m'=1}^{M} e_{m',m}$ and sends it to $\mathscr{M}_m$.
8: For $m \in [1:M]$, $\mathscr{M}_m$ decrypts $w_\ell = D_P(p_m, \mathsf{sk}_m)$.
9: For $m \in [1:M]$, $\mathscr{M}_m$ computes $y'_m = \prod_{r=\gamma_m+1}^{\gamma_{m+1}} \psi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$.
10: For $m, m' \in [1:M]$, each $\mathscr{M}_m$ computes $e'_{m,m'} = E_{EG}(y'_m, \mathsf{pk}'_{m'})$ and sends it to the server $S_{m'}$.
11: For $m \in [1:M]$, the server S_m computes $p'_m = \prod_{m'=1}^{M} e'_{m',m}$ and sends it to $\mathscr{M}_m$.
12: For $m \in [1:M]$, $\mathscr{M}_m$ decrypts $w'_\ell = D_{EG}(p'_m, \mathsf{sk}'_m)$.
13: **end for**
11: Every manager computes $\psi(w_L, w'_L) = \varphi(B(\mathscr{D}))$ and sends it to the client.

3.2 The PAA-HE protocol

The PAA-HE protocol is described in Algorithm 2. It combines the ElGamal and Paillier encryption schemes in one system. The managers set up their ElGamal and Paillier encryption schemes in line 1 of Algorithm 2. Each manager $\mathscr{M}_m$ sends public Paillier key pk_m and public ElGamal key pk'_m to all other managers.

Line 3 of Algorithm 2 initializes the values $w_0 = w_0 = 0$, required for iterations of the loop in lines 4 to 13. Each iteration of the loop assumes that, for $\ell \in [1:L]$, the values $w_{\ell-1}$ and $w'_{\ell-1}$ have already been determined in the previous iteration or in line 3. Each manager locally computes the subsum y_m in line 5 and uses the Paillier encryption to encrypt it and to send the encryption $e_{m,m'}$ to the corresponding server $S_{m'}$, for all $m' \in [1:M]$, line 6. The server computes the product $p_m = \prod_{m'=1}^{M} e_{m',m}$ and sends it to $\mathscr{M}_m$ in line 7. It follows from the homomorphic property (2) that p_m is an encryption of the sum $w_\ell = \sum_{r=1}^{R} \varphi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ in (7). Every manager $\mathscr{M}_m$ uses the Paillier scheme to decrypt w_ℓ in line 8. Each manager locally computes the subproduct y'_m in line 9 of Algorithm 2. The manager uses the ElGamal encryption scheme to encrypt y'_m using all public keys $\mathsf{pk}'_{m'}$ and to send the encryption $e'_{m,m'}$ to the corresponding server $S_{m'}$, for all $m' \in [1:M]$, line 10. The server computes the product $p'_m = \prod_{m'=1}^{M} e'_{m',m}$ and sends it to $\mathscr{M}_m$ in line 11. It follows from the homomorphic property (1) that p'_m is an encryption of the sum $w'_\ell = \prod_{r=1}^{R} \psi_\ell(\mathbf{v}_r, w_{\ell-1}, w'_{\ell-1})$ in (7). Each manager $\mathscr{M}_m$ uses the ElGamal scheme to decrypt w'_ℓ in line 12. It follows from (7) that $\psi(w_L, w'_L) = \varphi(B(\mathscr{D}))$. Each manager computes $\psi(w_L, w'_L)$ locally and sends it to the client in line 14 of Algorithms 2.

3.3 Theoretical analysis

For comparison, we include a direct application of the ElGamal and Paillier cryptosystems denoted by EGP. It transfers the required fields of all data vectors to the new servers in encrypted form and uses the homomorphic properties to perform the addition and multiplication of encrypted values without revealing their contents. The computation and communication complexities of PAA-SSS, PAA-HE, and EGP are presented in Table 2.

The security model considered in the present article assumes that all the managers $\mathscr{M}_1, \ldots, \mathscr{M}_M$ are honest, but

Table 2 Computation and communication complexities of the protocols

	Complexity	
Protocol	Computation	Communication
PAA-SSS	$O(M^2)$	$O(M^2)$
PAA-HE	$O(M)$	$O(M)$
EGP	$O(R)$	$O(RM)$

may be curious. This is a natural assumption, because the managers represent official organisations, which are not anonymous.

The servers $S_1, \ldots, S_M$ are new and are intended to process a lot of confidential information. They are likely to be targeted by the active outsider attackers. This is why our security model includes active outsider attackers capable of compromising some of the servers $S_1, \ldots, S_M$. This means that Byzantine faults may take place in the operation of the servers $S_1, \ldots, P_M$, when the faulty servers are trying to hide the fact that they have been compromised, but may output incorrect results of their calculations.

To concentrate on solving the new problem addressed in this paper, we assume that the communication between all participants is secure.

Theorem 1 *The PAA-SSS protocol produces correct answers to distributed queries of the class $\mathscr{C}$ if the active outsider attackers have compromised at most $\lceil M/3 \rceil - 1$ of the servers $S_1, \ldots, S_M$. The PAA-HE protocol produces correct answers to distributed queries of the class $\mathscr{C}$ if the active outsider attackers have compromised at most $\lceil M/2 \rceil - 1$ of the servers $S_1, \ldots, S_M$. Moreover, in both of these cases, the active outsider attackers cannot derive confidential information of individual managers by combining the data received from the compromised servers during the execution of each protocol.*

Proof First, suppose that the active outsider attackers have compromised at most $\lceil M/3 \rceil - 1$ of the servers $S_1, \ldots, S_M$. We are going to prove that the PAA-SSS protocol produces correct answers to distributed queries from the class $\mathscr{C}$, and that the active outsider attackers cannot derive confidential information of individual managers by combining the data available to them from the corresponding compromised servers.

Denote the number of the compromised servers $S_1, \ldots, S_M$ by $k \leq \lceil M/3 \rceil - 1$. It suffices to complete the proof in the most difficult case, where k is the largest integer with $k \leq \lceil M/3 \rceil - 1$. To simplify notation, we assume that $M = 3k + 1$.

It follows that in the set of M private shares $t_{\ell,m} = \Theta_m(z_{1,m}, \ldots, z_{M,m})$, for $m \in [1 : M]$, calculated in line 5 of Algorithm 1, at most k of these private shares may be compromised.

In line 5 of Algorithm 1, the manager $\mathscr{M}_m$ uses (4) to recover w_ℓ from

$$t_{\ell,1}, \ldots, t_{\ell,M}. \tag{13}$$

At most k of (13) may be incorrect. Since $t_{\ell,m} = \Theta_m(z_{1,m}, \ldots, z_{M,m})$, it follows from (5) that the secret shares encoding w_ℓ coincide with (13), which are equal to the values

$$f(\alpha^0), f(\alpha^1), \ldots, f(\alpha^{M-1}), \tag{14}$$

of the polynomial $f(x) = w_\ell + a_1 x + \cdots + a_k x^k$ encoding w_ℓ, as explained in Sect. 2.3.

Defining $a_{k+1} = a_{k+2} = \cdots = a_{M-1} = 0$, we get the sequence

$$a_0, a_1, \ldots, a_{M-1}. \tag{15}$$

Equation (5.182) in [22, Sect. 5.8.9] shows that (14) is a Discrete Fourier Transform of (15). The formula for the Reverse Fourier Transform (Equation (5.184) in [22, Sect. 5.8.9]) implies that $a_i = \frac{1}{M}\widehat{f}(\alpha^{-i})$, for all $i \in [0 : M-1]$, where

$$\widehat{f} = \Omega_1(w_\ell) + \Omega_2(w_\ell)x + \cdots + \Omega_M(w_\ell)x^{M-1}. \tag{16}$$

Therefore,

$$\widehat{f}(\alpha^{-i}) = 0, \text{ for } i \in [k+1 : M-1]. \tag{17}$$

Since $\alpha^M = 1$, we get $\alpha^{-i} = \alpha^{M-i}$ for $i \in [0 : M-1]$. If we substitute (16) into (17), then we get

$$\sum_{i=0}^{M-1} \alpha^{r \cdot i} \cdot \Omega_i(w_\ell) = 0 \text{ for } r \in [1 : 2k]. \tag{18}$$

It follows that

$$\alpha, \alpha^2, \ldots, \alpha^{2k} \tag{19}$$

are the roots of the polynomial (16). Since $\alpha^M = 1$ and $M = 3k + 1$, the set (19) is equal to the set

$$\alpha^{k+1}, \alpha^{k+2}, \ldots, \alpha^{M-1}. \tag{20}$$

Denote by $M_{\alpha^i}(x)$ the minimal polynomial of α^i. The above conditions prove that the polynomial

$$g(x) = \text{ lcm } \{M_{\alpha^{k+1}}(x), \ldots, M_{\alpha^{M-1}}(x)\} \tag{21}$$

divides (16). It follows that (13) is a codeword in the cyclic code of length M generated by $f(x)$. This means that this code is the BCH code of designed distance $2k + 1$ (see (5.105) in [22, Sect. 5.8.2]). As explained in [22, Sect. 5.8.4], it has an error-correcting algorithm, which corrects k errors. The manager can apply it and recover w_ℓ in line 6 of Algorithm 1.

Likewise, in line 9 of Algorithm 1, the manager $\mathscr{M}_m$ can use the BCH error-correction algorithm to correct all possible errors and recover w'_ℓ.

After all iterations of the loop in line 3 of Algorithm 1, the managers recover w_L and w'_L. Then every manager can locally compute the correct value $\psi(w_L, w'_L) = \varphi(B(\mathscr{D}))$.

Next, we prove that during steps of Algorithm 1 the active outsider attackers cannot derive confidential information concerning the data of separate managers by combining the values available to them from the

corresponding compromised servers. It was proved in [19] that, for $k \leq \lceil M/3 \rceil - 1$, if a secret value is considered as a uniformly distributed random variable over $\mathscr{F}$, then the values (3) are k-wise independent random variables that are uniformly distributed over $\mathscr{F}$, and therefore a set of k shares gained by the active outsider attackers from the k compromised servers cannot help to discover any confidential information.

In Algorithm 1, formula (3) is applied in lines 3 and 7 to communicate three secret values as sets of secret shares sent to the servers.

Therefore, it is impossible for the active outsider attackers to derive any confidential information from the values available to them as private shares from at most k compromised servers.

In line 4, each server $S_{m'}$ receives $z_{m,m'}$, which is calculated using (3). It follows that the values received by the compromised servers are k-wise independent random variables uniformly distributed over $\mathscr{F}$. Hence the active outsider attackers cannot use these values to derive confidential information.

In line 7, each server $S_{m'}$ receives $z'_{m,m'}$, which is calculated using (3). Therefore, the values received by the compromised servers are k-wise independent random variables uniformly distributed over $\mathscr{F}$. Thus, it is impossible for the active outsider attackers to deduce confidential information using these values.

Finally, in line 8 Algorithm 1 the servers use the procedure described in Sect. 2.3 for computing $t'_{\ell,m} = \Delta_m(z'_{1,m}, \ldots, z'_{M,m})$, for $m \in [1 : M]$. This procedure involves communicating data between the servers. The compromised servers receive new intermediate values as secret shares. However, the procedure uses randomization polynomials as explained in Sect. 2.3. It follows that the intermediate secret shares transferred to the compromised servers during this procedure also are k-wise independent random variables uniformly distributed over $\mathscr{F}$. Again, it is impossible for the active outsider attackers to deduce confidential information from the intermediate secret shares transferred to the compromised servers during the computation of the $t'_{\ell,m}$.

This proves that in Algorithm 1 the active outsider attackers cannot derive confidential information from the values they get from the compromised servers.

Second, suppose that the active outsider attackers compromised $k \leq \lceil M/2 \rceil - 1$ of the servers $S_1, \ldots, S_M$. It remains to prove that PAA-HE protocol produces correct answers to distributed queries from the class $\mathscr{C}$, and that the active outsider attackers cannot derive confidential information of individual managers by combining the data available to them from the compromised servers.

In Algorithm 2, the servers $S_1, \ldots, S_M$ perform identical computations. Evidently, $\lceil M/2 \rceil - 1$ is the largest integer that is strictly less than $M/2$. Since $k \leq \lceil M/2 \rceil - 1$, we get $k < M/2$. It follows that only the minority of the servers $S_1, \ldots, S_M$ are compromised, and so the managers can obtain correct results by using the majority of correct results received from uncompromised secure servers.

The servers $S_1, \ldots, S_M$ receive only encrypted values in Algorithm 2. They perform all calculations in encrypted form using the ElGamal and Paillier homomorphic properties. Since it is well known that the ElGamal and Paillier cryptosystems are secure, it follows that the active outsider attackers cannot derive any confidential information from the encrypted values available to them from the compromised servers. This completes the proof. □

4 Experimental set up and outcomes

This section is devoted to experiments using real datasets from the UCI Machine Learning Repository [23], the parameters of which are summarized in Table 3. To investigate the performance of our protocols for larger collections of data, we generated synthetic sets with the numbers of vectors ranging up to 10^9.

Our experiments investigate the effectiveness of the PAA-SSS and PAA-HE protocols comparing them with EGP protocol.

The PAA-SSS and PAA-HE protocols are the first privacy-preserving protocols for computing of distributed numerical queries over large distributed collections of data providing protection against active outsider attackers and minimizing the communication and computation costs for big data. Other protocols considered in the literature previously, cannot protect against active outsider attackers in the situation considered in the present paper, and so they cannot be included in our experiments.

Table 3 Datasets from the UCI Machine Learning Repository used in our experiments

Dataset	Number of instances
Liver disorders [24]	345
Thoracic surgery [25]	470
Cervical cancer [26]	858
Diabetic Retinopathy [27]	1151
Thyroid disease [28]	1600
Cardiotocography [29]	2126
Diabetes US hospitals [30]	3000

The experiments dealt with computing the geometric mean, it belongs to the difference of the two important classes $\mathscr{C} \setminus \mathscr{K}$, as explained in Sect. 3, and since it is an interesting statistic never considered in experimental studies in this research direction previously.

In our experiments, we included two values of the number M of the dataset managers: $M = 5$ and $M = 10$. Accordingly, every dataset was divided into $M = 5$ and $M = 10$ separate subsets of approximately equal size. A synthetic dataset with the number of vectors up to 10^9 and with normally distributed random values of confidential features was generated.

The communication time is proportional to the size of data transferred during the execution of the protocol divided by the speed of Internet transfer of data. This is why for comparing the communication costs of the protocols, our diagrams include the total size of data transferred in the experiments.

In this paper, our diagrams with the total communication costs contain only the volume of data communicated between the separate managers. The process of obtaining their original vectors from the corresponding local datasets is not a part of communication.

Since the communication time is determined by the size of data that has to be transferred in steps of the protocols and the bandwidth or speed of transfer of data over the Internet, we compare only the combined amount of data to be transferred to and from the servers $S_1, \ldots, S_M$.

The comparison of the performance of the algorithms are presented in Figs. 1, 2, 3, and 4. These outcomes show that both PAA-SSS and PAA-HE are much more efficient than the EGP protocol, and that PAA-HE outperforms all other protocols.

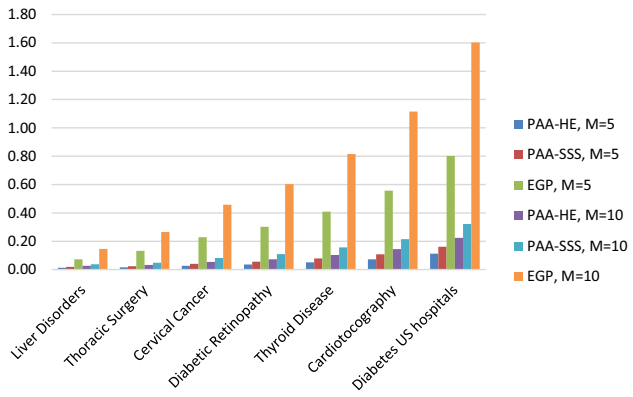


Fig. 1 The computation time (seconds) of PAA-HE, PAA-SSS, and EGP protocols with $M = 5$ and $M = 10$, for the data described in Table 3

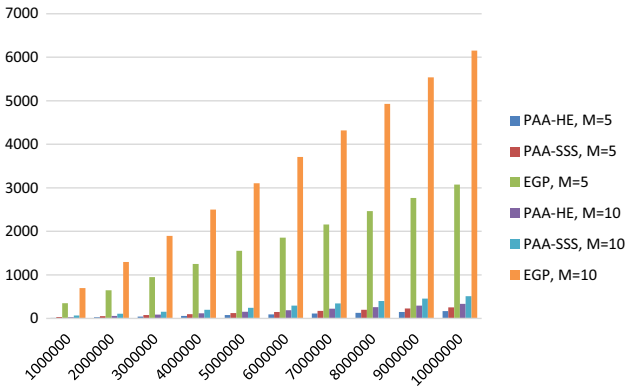


Fig. 2 The computation time (seconds) of PAA-HE, PAA-SSS, and EGP protocols with $M = 5$ and $M = 10$, for synthetic data

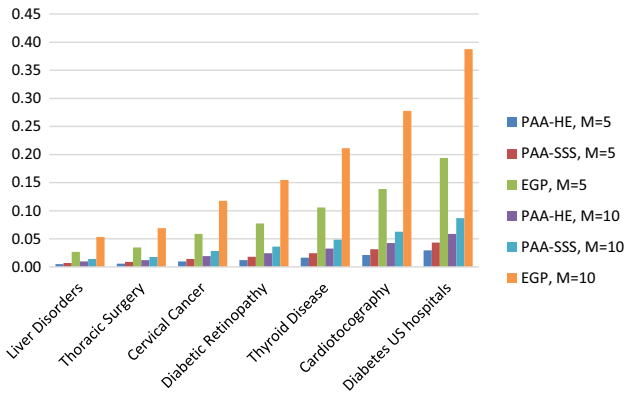


Fig. 3 The data (MB) communicated by PAA-HE, PAA-SSS, and EGP protocols with $M = 5$ and $M = 10$, for the data described in Table 3

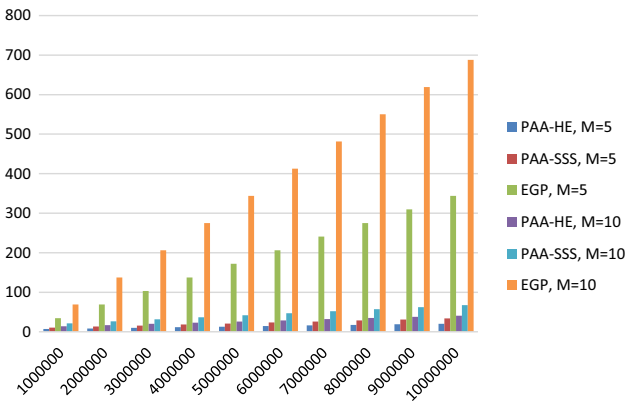


Fig. 4 The data (MB) communicated by PAA-HE, PAA-SSS, and EGP protocols with $M = 5$ and $M = 10$, for synthetic data

5 Conclusion

The development of privacy-enhancing techniques and protocols for data aggregation and analytics in wireless networks requires novel methods for efficient and privacy-

preserving computation of distributed queries with the protection of outcomes from active attackers.

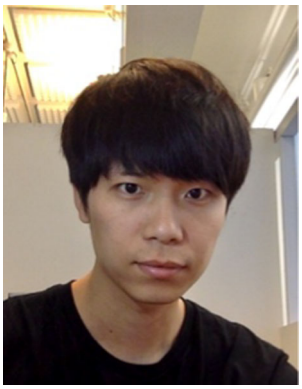
In this paper, we propose two protocols for the protection of confidential data from active outsider attackers in this situation: PAA-SSS and PAA-HE. The analysis and experimental outcomes demonstrate that PAA-SSS and PAA-HE are more efficient than alternative options.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

References

1. Brito, C., Esteves, M., Peixoto, H., Abelha, A., Machado, J. (2019). A data mining approach to classify serum creatinine values in patients undergoing continuous ambulatory peritoneal dialysis. *Wireless Networks*, pp. 1–9.
2. Jeon, D., Tak, B.: Blackeye: automatic IP blacklisting using machine learning from security logs. *Wireless Networks*, pp. 1–12 (2019)
3. Fernandes, D., Ferreira, A. G., Abrishambaf, R., Mendes, J., & Cabral, J. (2021). A machine learning-based dynamic link power control in wearable sensing devices. *Wireless Networks, 27*(3), 1835–1848.
4. Yang, X., & Kuang, L. (2021). Social media data mining and knowledge discovery under wireless network. *Wireless Networks, 27*(5), 3375–3376.
5. Ren, Y., Li, X., Sun, S.-F., Yuan, X., & Zhang, X. (2021). Privacy-preserving batch verification signature scheme based on blockchain for vehicular ad-hoc networks. *Journal of Information Security and Applications, 58*, 102698.
6. Sun, S.-F., Steinfeld, R., Lai, S., Yuan, X., Sakzad, A., Liu, J.K., Nepal, S., Gu, D. (2021). Practical non-interactive searchable encryption with forward and backward privacy. In *NDSS*.
7. Guo, Y., Wang, M., Wang, C., Yuan, X., & Jia, X. (2020). Privacy-preserving packet header checking over in-the-cloud middleboxes. *IEEE Internet of Things Journal, 7*(6), 5359–5370.
8. Xu, W., Zhao, Q., Zhan, Y., Wang, B., & Hu, Y. (2022). Privacy-preserving association rule mining based on electronic medical system. *Wireless Networks, 28*(1), 303–317.
9. Lin, L., Liu, T., Hu, J., & Ni, J. (2016). Pqsel: Combining privacy with quality of service in cloud service selection. *International Journal of Big Data Intelligence, 3*(3), 202–214.
10. Demir, S., & Tugrul, B. (2018). Privacy-preserving trend surface analysis on partitioned data. *Knowledge-Based Systems, 144*, 16–20.
11. Ta, H. Q., Pham, Q.-V., Ho-Van, K., & Kim, S. W. (2022). Covert communication with noise and channel uncertainties. *Wireless Networks, 28*(1), 161–172.
12. Dobrilović, D., Brtka, V., Jotanović, G., Stojanov, Ž, Jauševac, G., & Malić, M. (2022). The urban traffic noise monitoring system based on Lorawan technology. *Wireless Networks, 28*(1), 441–458.
13. Abadleh, A., Al-Mahadeen, B. M., AlNaimat, R. M., & Lasassmeh, O. (2021). Noise segmentation for step detection and distance estimation using smartphone sensor data. *Wireless Networks, 27*(4), 2337–2346.
14. Son, H. (2021). Cooperative beamforming based artificial noise in uplink wiretap channels. *Wireless Networks, 27*(3), 1861–1874.
15. Kelarev, A., Yi, X., Badsha, S., Yang, X., Rylands, L., Seberry, J. (2019). A multistage protocol for aggregated queries in distributed cloud databases with privacy protection. Future Generation Computer Systems, pp. 368–380 (90)
16. Yi, X., Paulet, R., & Bertino, E. (2014). *Homomorphic encryption and applications*. New York: Springer.
17. ElGamal, T. (1985). A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory, 31*, 469–472.
18. Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pp. 223–238.
19. Shamir, A. (1979). How to share a secret. *Communications of the ACM, 22*, 612–613.
20. Ben-Or, M., Goldwasser, S., Wigderson, A. (1988). Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of 20th annual ACM symposium on theory of computing, STOC'88*. ACM, pp. 1–10.
21. NIST/SEMATECH (2019). E-Handbook of Statistical Methods. Available at http://www.itl.nist.gov/div898/handbook/, viewed 5 May 2019.
22. Borda, M. (2011). *Fundamentals in information theory and coding*. Berlin: Springer.
23. Lichman, M. (2018). UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, available at http://archive.ics.uci.edu/ml, viewed 25 August 2017.
24. McDermott, J., & Forsyth, R. S. (2016). Diagnosing a disorder in a classification benchmark. *Pattern Recognition Letters, 73*, 41–43.
25. Zieba, M., Tomczak, J. M., Lubicz, M., & Swiatek, J. (2013). Boosted SVM for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients. *Applied Soft Computing, 14*, 99–108.
26. Fernandes, K., Cardoso, J.S., Fernandes, J. (2017). Transfer learning with partial observability applied to cervical cancer screening. In *Iberian Conference Pattern Recognition and Image Analysis, IbPRIA 2017*. LNCS, vol.10255, pp. 243–250.
27. Antal, B., & Hajdu, A. (2014). An ensemble-based system for automatic screening of diabetic retinopathy. *Knowledge-Based Systems, 60*, 20–27.
28. Quinlan, J. R., Compton, P. J., Horn, K. A., Lazurus, L. (1986). Inductive knowledge acquisition: A case study. In *Proceedings of 2nd Australian conference applications of expert systems. Sydney, Australia*, pp. 137–156.

29. Ayres-de Campos, D., Bernardes, J., Garrido, A., Marques-de-Sa, J., Pereira-Leite, L. (2000). SisPorto 2.0: A program for automated analysis of cardiotocograms. *Journal of Maternal-Fetal Medicine, 5*, pp. 311–318.
30. Strack, B., DeShazo, J. P., Gennings, C., Olmo, J. L., Ventura, S., Cios, K. J., & Clore, J. N. (2014). Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records. *BioMed Research International, 2014,* 1–11.



Xuechao Yang received the bachelor's degree in information technology and the Bachelor of Computer Science degree (Hons.) from RMIT University, Melbourne, VIC, Australia, in 2013 and 2014, respectively, and the Ph.D. degree from the School of Science, RMIT, with data61, CSIRO in 2018. He is a Research Fellow with the School of Computing Technologies, RMIT University. His research interests include cryptosystems, privacy preserving, and blockchain technology.

Andrei Kelarev is a Research Fellow in the School of Computing Technology, RMIT University, Australia. He is an author of two books and 198 journal articles. He worked as an Associate Professor in the University of Wisconsin and University of Nebraska in USA, a Senior Lecturer in the University of Tasmania in Australia, and was a Chief Investigator of a large Discovery grant from Australian Research Council. He is working on cyber security applications of machine learning and data mining.

Xun Yi received the Ph.D. degree from Xidian University, Xi'an, China. He is currently a Professor with the School of Computing Technologies, RMIT University, Melbourne, VIC, Australia. He has published over 160 research papers in international journals and conference proceedings. His research interests include applied cryptography, computer and network security, and privacy-preserving data mining. Prof. Yi has been an Associate Editor for IEEE Transaction Dependable and Secure Computing since 2014. He has ever undertaken program committee members for over 30 international conferences.