Data & Knowledge Engineering 68 (2009) 1224–1236

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Accurate and large-scale privacy-preserving data mining using the election paradigm

Emmanouil Magkos [a,*], Manolis Maragoudakis [b], Vassilis Chrissikopoulos [a], Stefanos Gritzalis [b]

[a] *Department of Informatics, Ionian University, Plateia Tsirigoti 7, Corfu 49100, Greece*
[b] *Department of Information and Communication Systems Engineering, University of the Aegean, Karlovassi, Samos, Greece*

ARTICLE INFO

Article history:
Available online 23 June 2009

Keywords:
Security and privacy
Distributed data mining
Homomorphic encryption
Random Forests classification

ABSTRACT

With the proliferation of the Web and ICT technologies there have been concerns about the handling and use of sensitive information by data mining systems. Recent research has focused on distributed environments where the participants in the system may also be mutually mistrustful. In this paper we discuss the design and security requirements for large-scale privacy-preserving data mining (PPDM) systems in a fully distributed setting, where each client possesses its own records of private data. To this end we argue in favor of using some well-known cryptographic primitives, borrowed from the literature on Internet elections. More specifically, our framework is based on the classical homomorphic election model, and particularly on an extension for supporting multi-candidate elections. We also review a recent scheme [Z. Yang, S. Zhong, R.N. Wright, Privacy-preserving classification of customer data without loss of accuracy, in: SDM' 2005 SIAM International Conference on Data Mining, 2005] which was the first scheme that used the homomorphic encryption primitive for PPDM in the fully distributed setting. Finally, we show how our approach can be used as a building block to obtain Random Forests classification with enhanced prediction performance.

1. Introduction

Data mining technologies, aiming at extracting valuable, non obvious information from large quantities of data [2], have broad applications in areas related to market research, as well as to financial and scientific research. Despite the potentials for offering valuable services, there have been concerns about the handling and use of sensitive information by data mining systems [3,4]. The problem is even more intense nowadays with the proliferation of the Web and ICT technologies, and the progress in network, storage and processor capacity, where an enormous pool of sensitive digital data can be easily gathered, or inferred from massive collections of public data such as in social networks, by using well-known data mining techniques. Even when access to sensitive data is controlled, public data can sometimes be used as a path towards private data [4].

Traditionally, data mining algorithms have been applied in centralized collections of data. Admittedly, the accuracy of any data mining task may be also increased when data are collected from various locations. With *distributed databases*, data may be horizontally or vertically partitioned among a set of sites, where each site may hold similar data about different people or different data about the same set of people, respectively. This is also known as the *Server to Server* (S2S) setting [5]. In a fully distributed setting, also known as *Client to Server* (C2S) [5], customers may be on hold of their own collections of sensitive data. Such data may need to be correlated with other clients' data, for example in order to provide some useful service. From

[*] Corresponding author.
E-mail addresses: emagos@ionio.gr (E. Magkos), mmarag@aegean.gr (M. Maragoudakis), vchris@ionio.gr (V. Chrissikopoulos), sgritz@aegean.gr (S. Gritzalis).

0169-023X/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2009.06.003

a security point of view we note that, in both settings, sites and clients may be mutually mistrustful. As a result, the traditional warehousing approach, where dispersed data are gathered into a central site for building the mining model, raises privacy concerns. The trivial approach of performing data mining at each site independently and then combine the results (e.g., [6]) cannot always be accurate [7,8] or even possible (e.g., in the *C2S* setting).

Furthermore, organizations and people are often reluctant to share their private data due to law, compliance, ethics, commercial or other reasons (e.g., see [8–11]). Especially in distributed *statistical databases*, where there is a need to extract statistical information (e.g., sum, count, average, entropy, information gain, etc.) without compromising the privacy of the individuals [12], the need for privacy can also be viewed from a different angle: it would enable collaboration between data holders (e.g., customers, organizations), if they were assured that their sensitive information would be protected. As a result, privacy concerns may prevent improving the accuracy of data mining models. To this end, *privacy-preserving data mining* (PPDM) has been evolved as a new branch of research [13].

Traditionally, the use of cryptographic primitives has also been well studied by the database security community [14]. In the PPDM setting, recent results have exemplified the inherent *trade-off* between the privacy of input data and the accuracy of the data mining task. Contrary to other strategies, modern cryptographic mechanisms usually do not pose such dilemmas and offer formal definitions, clearly stated assumptions and rigorous proofs of security. Frequently however, such mechanisms may be rather inefficient. In the distributed setting for example, where sites may also be mutually mistrustful, much work has been done in the *S2S* setting using a *collaborative approach* [15] that fits better with small-scale systems. In the *C2S* setting however, there is a need for solutions that are both efficient and rigorous. Challenges increase when we also desire scalability, where a large number of clients (e.g., thousands or millions) may participate in the mining process.

1.1. Our contribution

In this paper we work at a high level and discuss design and security requirements for large-scale PPDM in the *C2S* setting. We also explore whether it is possible to use efficient cryptography at the application layer to fulfill our requirements and perform PPDM in statistical databases, while maintaining the accuracy of the results. To this end we argue in favor of borrowing knowledge from a broad cryptographic literature for Internet elections. We discuss similarities and differences with our system requirements and refer to generic privacy-preserving models for large-scale elections. We argue in favor of using a variation of the classical homomorphic model [16] as a framework for large-scale PPDM in *C2S* statistical databases. More specifically we consider some recent extensions of the classical model, proposed in [17,18], for multi-candidate selections (e.g., *1-out-of-ℓ* and *k-out-of-ℓ* selections). We give a sketch of a simple frequency miner on a large set of fully distributed customer databases and discuss its security, efficiency, and possible extensions. Furthermore, we discuss some weaknesses and describe an attack on a recent scheme of Yang et al. [1] which was the first work that used the homomorphic encryption primitive [16] for PPDM in the *C2S* setting. Finally, we use our PPDM approach as a building block to obtain a Random Forests classifier over a set of homogeneous databases with horizontally-partitioned data, and present experimental results. The preliminary idea of this work has been published in [19].

2. Related work

A very common approach in the PPDM literature has been *data perturbation*, where original data are perturbed and the data mining model is built on the randomized data. For example, data perturbation has been used for classification [20] and building association rules [21,22]. Typically, this approach is very efficient but involves a trade-off between two conflicting requirements: the privacy of the individual data and the accuracy of the extracted results [12,23–25]. In addition, there are cases where the disclosure of some randomized data about a client may reveal a certain property of the client's private information, a situation known as a *privacy breach* [21,25].

Following the line of work that begun with Yao's generic secure two-party computation [26], and extended to general-purpose *Secure Multiparty Computation* (SMC) (e.g. [15]), most proposals in the cryptographic literature for PPDM are based on what is also known as the *collaborative approach* (e.g. [3,7,27–29,10]). This approach involves special-purpose protocols that belong to the SMC family of privacy-preserving protocols. These are interactive protocols, run in a distributed network by a set of entities with private inputs, who wish to compute a function of their inputs in a privacy-preserving manner; privacy here means that no more information is revealed to an entity than can be inferred from its own input and output, and from the output of the joint computation [30]. The collaborative approach has been used for mining association rules on both horizontally [27] and vertically partitioned databases (e.g., [7,29]). Classification models that use this approach involve decision trees [3,31], naive Bayes classification for horizontally-partitioned data (e.g., [8,10]), as well as decision trees for vertically partitioned data [32] (also see [33] for related work in this area). Collaborative protocols inherently fit to small-scale systems as they require multiple communication rounds among the participants and are efficient as long as the number of participants is kept small. With larger scales and data sets, privacy usually comes at a high performance cost [31].

Another cryptographic tool that has been used in the PPDM literature is the *homomorphic encryption* primitive that exploits an interesting algebraic property of several encryption functions, where for example there is an operation $\oplus$ defined on the message space and an operation $\otimes$ defined on the cipher space, such that the "product" of the encryptions of any two private inputs equals the encryption of the "sum" of the inputs:

$$E(M_1) \otimes E(M_2) = E(M_1 \oplus M_2) \tag{1}$$

Although the primitive has been used in some recent schemes for collaborative data mining [28,29,34,10], none of these approaches can be used in the *C2S* setting for large-scale PPDM. To this end, in this paper we will discuss the use of the homomorphic encryption primitive in view of the design and security requirements that will be defined in Section 3. Very close to our research has been the work of Yang et al. [1] which, to our best of knowledge, was the first scheme that used the homomorphic encryption primitive in order to build a privacy-preserving frequency mining algorithm in the *C2S* setting. The algorithm is then used in [1] as a building block to design a protocol for naive Bayes learning. The authors in [1] also discuss applications to other data mining techniques such as decision trees and association rule mining. We will review the scheme of [1] separately in Section 5.

Finally, we consider as closely related but out of scope the extensive literature on access control and general-purpose security for database systems [12,4], such as authorization and information flow control (e.g., multilevel and multilateral security [35]), audit policies, trusted platforms, query restriction and inference control policies [12,36–38], or anonymization techniques [39]. We consider these areas as orthogonal and rather concerning aggregate data in more or less controlled environments.

3. Design and security requirements for PPDM in the C2S setting

3.1. Design requirements

We consider efficiency and scalability in distributed statistical databases:

1. *Statistical databases:* We consider data mining systems that extract statistical information (e.g., sum, count, average) from distributed databases.
2. *C2S setting:* The transaction database is horizontally partitioned and each client is on hold of its own collections of personal data. A third-party server (Miner) or a coalition of third-party servers (Miners) gather client-submitted encrypted messages and execute a function on these messages. At the simplest case, each message is a selection out of ℓ possible choices, where $\ell \geqslant 2$.
3. *Large-scale systems*: We consider a very large-scale setting where hundreds or even thousands of clients may participate in the system.
4. *Efficient systems*: We require computation and communication efficiency for both clients and miners. No inter-client communication is allowed, and one flow of data is allowed from each client to the Miner(s).

3.2. Threat model

At a high level, adversaries in distributed systems for data mining can be seen as either *semi-honest* (also referred to as *honest but curious*) or *malicious* [31]. In our threat model we consider both kinds of adversaries. Semi-honest adversaries are legal participants that follow the protocol specification. They behave the way they are supposed to, they do not collude or sabotage the process, but instead try to learn additional information given all the messages that were exchanged during the protocol. We note that the assumption of a semi-honest adversary has also been seen as a strong assumption (e.g., [40]).

In the malicious setting, we discriminate between an *internal* adversary *Int* and an *external* adversary *Ext*. *Int* will arbitrarily deviate from the protocol specification [31]. As a client, for example, *Int* will submit data that are not of the correct structure, in order to distort the system or infer private information. As a miner, *Int* will try to decrypt partial results and violate the privacy of a (set of) participant(s). We also assume that *Ext* will attempt to impersonate authorized users and submit forged messages to the system. In our threat model however, we assume that *Ext* cannot coerce honest clients to reveal their private inputs or compromise more than t miners, where t is an appropriate security parameter. Furthermore, we permit collusion of up to a certain number $(t-1)$ of (compromised) miners.

3.3. Security requirements

In view of our threat model, we consider a set of basic non-functional requirements for secure and large-scale PPDM in the *C2S* setting.

1. *Eligibility*: Only eligible clients are able to submit a message to the mining system.
2. *Accuracy*: Input messages cannot be altered, duplicated or eliminated from the system.
3. *Privacy*: A crucial aspect of privacy is the *unlinkability* between input data and the identity of the client who submits it. Another related criterion is *secrecy*: Atomic input data remain secret and only an aggregate of the client-submitted data is to be revealed.
4. *Verifiability*: All clients are able to verify that the output of the mining system is a correct function of the submitted messages.

5. *Robustness*: The system is secure despite any failure or malicious behaviour by a (reasonably sized) coalition of clients, miners or outsiders.

Note 1. Stronger notions of privacy, namely *receipt freeness* or *uncoercibility* [41,42] could also be introduced in PDDM, under the malicious threat model. Receipt freeness dictates that no client should be able to prove her input to others (even if she wants to). With uncoercibility, no party should be able to coerce a client into revealing her input. While in this paper we do not consider such threats, we take the occasion to note that future research may also consider stronger definitions for PPDM in the malicious threat model.

3.4. Basic building blocks

3.4.1. Threshold cryptosystems

Threshold cryptography [43] has been proposed to establish robustness in distributed protocols. At a high level and for any (n, t) threshold cryptosystem, the decryption power can be divided among a set of n independent miners as follows: First, a *key generation* protocol is executed so that the n miners possess shares of the private decryption key that corresponds to the public encryption key of the system. Second, a *decryption protocol* where a subset of $t \leqslant n$ honest miners cooperatively decrypt a message, without reconstructing the private key. The miners may also prove, in zero-knowledge, correctness of their computation, so that the protocol is robust against any coalition of less than t malicious miners.

3.4.2. Zero-knowledge (ZK) proofs

ZK proofs are prover–verifier interactive protocols, where the prover proves a statement and the verifier learns nothing from the prover that he could not learn by himself, apart from the fact that the prover knows the proof [44]. ZK proofs will result in slower computation either at the client or at the miner side, depending on where they are used. In normal operation, they take the form of interactive protocols, however, it is also possible to transform these proofs into non-interactive proofs (public signatures) [45].

3.4.3. Bulletin boards

A bulletin board was introduced in [46] to allow authenticated communication between pairs of processes in a distributed system. All communication supported by the board is authenticated, each process is assumed to have its exclusive part on the board and messages cannot be erased or tampered with. All public data pertaining to the system (e.g., public keys of miners), may also be published on the board. Boards can be implemented based on an existing PKI and on replicated servers (to deal with Denial-Of-Service attacks). An implementation of the primitive was proposed in [47].

4. PPDM in view of the election paradigm

We argue that research for large-scale PPDM in the C2S setting could borrow knowledge from the vast body of literature on Internet voting systems [48]. These systems are not strictly related to data mining but they exemplify some of the difficulties of the multiparty case. Such systems fall, to a large extent, within our design requirements and also tend to balance well the efficiency and security criteria, in order to be implementable in medium to large-scale environments. In an Internet election for example, an election authority receives several encrypted *1-out-of-2* votes (e.g., either $Yes = 1$ or $No = -1$) and declares the winning candidate. In this setting the goal is to protect the privacy of the voters (i.e., unlinkability between the identity of the voter and the vote that has been cast), while also establishing eligibility of the voters, accuracy and verifiability for the election result.

4.1. Unlinkability in the homomorphic election model

This model is a general framework for the usage of any encryption scheme with specific algebraic properties in order to protect the privacy of the encrypted votes and establish accuracy and verifiability of the decrypted results. The encryption scheme has to be *randomized*, to preclude chosen-plaintext [49] attacks on the published encryptions. If this is the case then, the properties of the function in Eq. (1) allow either to tally votes as aggregates or to combine shares of votes (e.g., [50,51]), without decrypting single votes.

4.1.1. The classical election model of Cramer et al. [16]

In [16] votes are encrypted using a variation of ElGamal encryption [52] with addition as group operation of the message space. More specifically, there is a set of n authorities that announce on a bulletin board the election's public encryption key h, according to a threshold version of the ElGamal cryptosystem [16]. Each voter (from a set of c voters) chooses a random number $a \in Z_q$ and encrypts a vote $m_i \in \{1, -1\}$ as the pair $(x, y) = (g^a, h^a G^{m_i})$, where $h = g^s$ is the public key, g, G are generators of G_q and all the operations are modulo p. The voter then prepares a ZK proof [16] that the vote is valid, i.e., that $m_i \in \{-1, +1\}$ without revealing m_i; Otherwise it would be easy for a malicious voter to manipulate the final tally. The voter then signs and publishes the encrypted vote and the proof of validity on the bulletin board. At the end of the voting period, the authorities check the proofs, and execute a threshold protocol to jointly compute $G^T = Y/X^s$, where Y is the product of all y's, X is the product of all x's and T is the result, computed from $O(c)$ modular multiplications [16]. In the above scheme, voter privacy is reduced to the *Decisional Diffie Hellman* (DDH) assumption (the reader may refer to [16] for formal security arguments).

The homomorphic model does not require interactions between clients, and only one flow of data is sent to the server. Privacy is established in a strong cryptographic sense and the homomorphic property of the encryption scheme makes easy to establish universal verifiability for the final results, simply by performing a multiplication of the encrypted inputs and comparing the encrypted aggregate to the value published on the board. Robustness and fault-tolerance are ensured with threshold decryption, where a set of honest authorities cooperate and decrypt the encrypted aggregate.

4.2. Other generic models for unlinkability

In the election literature, it is interesting to point out the existence of two other generic models for establishing unlinkability [53]. In the *Mix-net model* (e.g., [42]), encrypted votes are shuffled (e.g., re-randomization and re-ordering of the votes) by a set of mix servers who prove, in ZK, the correctness of their computations. Finally, in the *blind signature* model [54], unlinkability is established at the vote preparation stage: the voter proves eligibility to vote and then submits a blinded (i.e., randomized) version of the encrypted vote to an election authority for validation. Later, this blinding is easily removed by the voter and a validated vote is submitted to the authority, using an anonymous channel.

Generic models for privacy-preserving elections have their pros and cons [53], in view of the design and security requirements. For example, the homomorphic model allows for *Yes/No* voting and supports efficient tallying, whereas the Mix-net model allows for plaintext votes but is less efficient in the tallying stage, because of the ZK proofs of correct mixing.

4.3. PPDM and the homomorphic (election) model

In the PPDM context, the homomorphic model fits better into our design and security requirements. We note that in the semi-honest model there may be no need for clients to construct complex ZK proofs on the correctness of their inputs. If we consider the malicious model, there have been some efficient constructions for ZK proofs of validity (e.g., [17,18]). In addition, the *universal verifiability* requirement in online elections, can also be relaxed and replaced with a requirement for *atomic verifiability*, where only participants in the protocol are able to verify the accuracy of the results. So we may be able to construct and choose among lightweight versions of some well-known cryptographic schemes that follow the homomorphic model, and adopt them to our PPDM setting. In Section 6 we will describe a simple scheme for PPDM, based on the classical homomorphic model.

5. Review of the Yang et al. [1] scheme

In [1] a fully distributed setting is considered, where the clients' database is horizontally partitioned, and every client possesses his own data. We briefly describe the PPDM protocol of [1], where a miner mines a large number of client data sets to compute frequencies of values. Let G be a group where the Discrete Logarithm problem is hard. All operations are done *modp*, where p is a publicly known and sufficiently large prime number. In a system with c clients, each client possesses two pairs of keys: $(x_i, X_i = g^{x_i})$, $(y_i, Y_i = g^{y_i})$, with g being a (publicly known) generator of the group G. Each client U_i knows his private keys (x_i, y_i), with values (X_i, Y_i) being the corresponding public keys. Furthermore, the protocol requires all clients to know the values X and Y, where $X = \prod_{i=1}^{c} X_i$, and $Y = \prod_{i=1}^{c} Y_i$. Each client is able to give a *Yes/No* answer d_i (where $Yes = 1$ and $No = 0$) to any question posed by the miner and the miner's goal is to learn $\sum_{i=1}^{c} d_i$. In the protocol, depicted in Fig. 1, all cli-

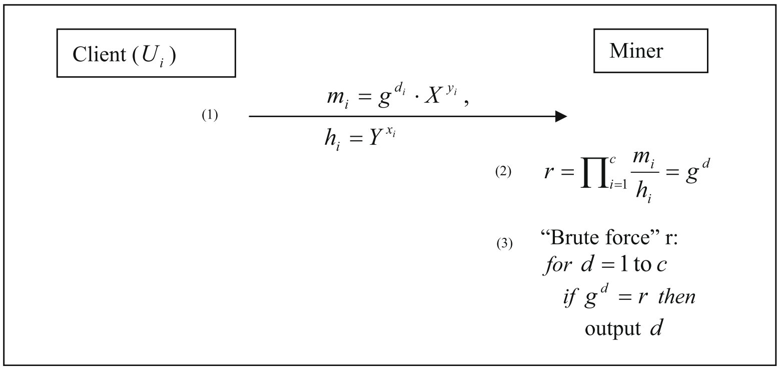


Fig. 1. A schematic representation of the protocol in [1].

ents in the system use a variant of the ElGamal encryption scheme [52]. For correctness and privacy analysis, please refer to [1].

Observe that in the scheme of [1], as well as in the scheme of [16], the computation of the tally (i.e., the result d that equals the sum of the plaintext inputs) involves a brute-force attack on the value $g^d \; modp$, and specifically $O(\sqrt(c))$ exponentiations in order to find the discrete logarithm. This stands because there are no trapdoors to compute d from g^d in ElGamal variants. In settings with only two candidates (e.g., *yes/no*) this is a relatively easy computation, at least for a moderately sized number of clients. For *1-out-of-*ℓ selections in a large-scale system with say c clients, the time complexity for decrypting the result would be $O(\sqrt(c)^{\ell-1})$ exponentiations, which is prohibitively expensive when ℓ is large [16,18].

We briefly describe two issues concerning the protocol. The first one refers to the need that each client must choose new x_i and y_i values after each execution of the protocol. This is actually a requirement in every randomized encryption scheme, where new randomness is used to increase the cryptographic security of the encryption process against chosen-plaintext attacks [49]. For example, in Fig. 1, if the client U_i uses the same x_i and y_i values during two successive runs, it will be trivial for an attacker (in the semi-honest model) to find out U_i's answers by trial and error. The above issue cannot be considered as an attack, since the authors in [1] write a remark about the need for updating the (x_i, y_i) values. However we rather consider this as a scalability issue: Prior to the execution of each run of the protocol (e.g., possibly during a key setup phase) each client must obtain or compute the numbers X and Y which are functions of the public keys (X_i, Y_i) of all system clients. In a fully distributed and large-scale scenario, where a very large number of system clients hold their own records, it may be difficult to pre-compute and/or publicize these values, turning the key setup phase into a complex procedure, especially in cases where the number of participants is not constant through different runs of the system.

In [19] we also argue that a single client may be able to disrupt the system. Indeed, in a system with say three clients U_1, U_2, U_3, let us assume that U_2 does not send her input, because of a system crash. Then the protocol executes as in Fig. 2 and a result cannot be found. One could argue that in the semi-honest threat model, all clients will adhere to the protocol specification and will not abstain from the protocol, however this may be considered as a strong assumption, especially in large-scale protocols (e.g., 10,000 clients in the experimental results in [1]). Furthermore, the semi-honest model does not preclude non-malicious system crashes or network failures. Observe that a client does not know a priori who will participate in the protocol, so the obvious fix of constructing the values X and Y as a function of the number of active participants will not work.

6. A generic PPDM approach for large-scale *C2S* systems

For large-scale PPDM in the *C2S* setting we adopt a variation of the classical homomorphic model. More specifically, our approach will be based on some efficient extensions of the homomorphic model, where *1-out-of-*ℓ or *k-out-of-*ℓ selections are allowed (e.g., [17,18]).

6.1. Background

In the usual (*1-out-of-2*) setting (e.g., either $Yes = 1$ or $No = -1$), a client who does not want to participate may give false information. Or, in the fully distributed setting where each client retains control of his transactions, the client may decide not to participate, although we may consider this as a privacy violation. As a result, the null input should also be considered in distributed mining protocols. Furthermore, knowledge cannot always be represented with a *Yes/No* decision. For example, a client may have to answer which group (e.g., among ℓ age groups) her age belongs to. As a result we are interested in multicandidate schemes, where in the simplest *1-out-of-*ℓ case each client makes one selection out of ℓ candidates and sends this privately to a frequency miner, as shown in Fig. 3.

Multi-candidate protocols have been first investigated in [50] and further studied in [16], where the computation of the final tally grows exponentially with the number ℓ of candidates. Baudron et al. [17] proposed the use of the *Paillier cryptosystem* [55] for conducting homomorphic elections with multiple candidates. The Paillier scheme provides a trapdoor to directly compute the discrete logarithm, thus making the computation of the tally efficient, even for large values of c and ℓ. They also presented a threshold version of the Paillier cryptosystem, as well as a ZK proof of validity for an encrypted message. We briefly recall the Paillier cryptosystem, leaving out some details on key generation and decryption [55]. Let $N = pq$

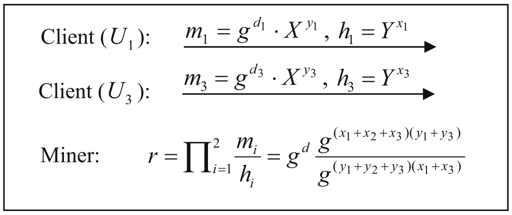
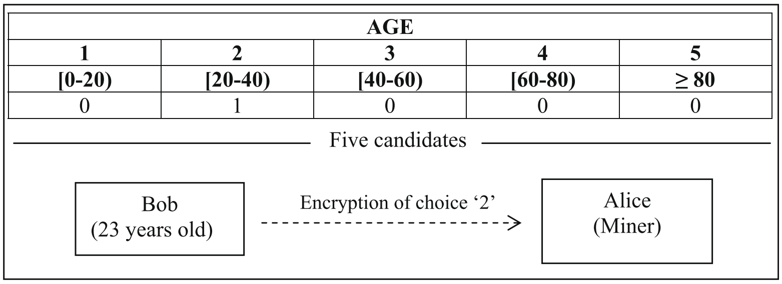

Client (U_1): $m_1 = g^{d_1} \cdot X^{y_1}, \; h_1 = Y^{x_1}$
Client (U_3): $m_3 = g^{d_3} \cdot X^{y_3}, \; h_3 = Y^{x_3}$
Miner: $r = \prod_{i=1}^{2} \frac{m_i}{h_i} = g^d \frac{g^{(x_1+x_2+x_3)(y_1+y_3)}}{g^{(y_1+y_2+y_3)(x_1+x_3)}}$

Fig. 2. A run with two active clients in a system with three registered clients.



AGE				
1	2	3	4	5
[0-20)	[20-40)	[40-60)	[60-80)	≥ 80
0	1	0	0	0

Fig. 3. A multi-candidate setting with *1-out-of-5* choices.

be an RSA modulus where p and q are two large primes, and g be an integer of suitable order modulo N^2. The public key is (N, g) while the secret key is the pair (p, q). To encrypt a message $m \in Z_n$, choose a random $x \in Z_n$ and compute $E(m) = g^m x^N (modN^2)$. The knowledge of the trapdoor (p, q) allows to efficiently decrypt and determine M. The Paillier cryptosystem has the homomorphic encryption property as in Eq. (1).

6.2. Assumptions

There are c clients and n miners in the system. We assume the existence of a randomized public key scheme with the homomorphic encryption property (e.g., the Paillier scheme). All questions to a database can be reduced to a set of *1-out-of-ℓ* answers. For example in [17] all answers are choices from the set $(1, M, M^2, \ldots, M^{\ell-1})$, with M being an integer larger than the number c of clients (e.g., $M = 2^{\log_2 c}$). For verifiability, all entities have access to a bulletin board. A key generation protocol has already taken place and the system's parameters and the public encryption key of the miners is published on the bulletin board, while the miners possess the matching shares of the private decryption key, according to a threshold version of the cryptosystem.

6.3. A simple frequency miner

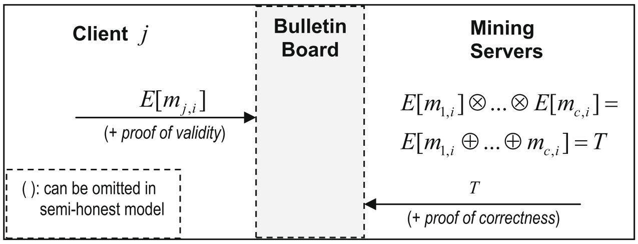
The steps of the simple protocol, depicted in Fig. 4:

1. A client j who wishes to select the ith answer, encrypts a message with the public key of the election authorities. For instance, using Paillier:

$$E[m_{j,i}] = g^{M^i} x^N (modN^2) \tag{2}$$

   For eligibility the client signs $E[m_{j,i}]$. Depending on the threat model, the client may also construct a proof of validity (e.g., the one proposed in [17] for Paillier encryption), and turn it into a non-interactive proof using the Fiat–Shamir heuristic [45]. The client then publishes the signature and proof on the bulletin board.
2. When a deadline is reached, the miners check the proofs of validity and compute the homomorphic “product” of the encrypted messages. The miners cooperate to decrypt the tally using threshold decryption. For instance, the Paillier-encrypted tally can be written in M-ary notation: $T = a_0 M^0 + a_1 M^1 + \ldots + a_{\ell-1} M^{\ell-1} (modN)$, which will directly reveal all a_i's, where $0 \leqslant a_i \leqslant c$ is the number of selections for answer i. The miners publish the encrypted product and final results on the board.

We stress that the above protocol is a sketch of a generic PPDM scheme in the fully distributed setting. Without loss of generality we adopted the notation of [17] for *1-out-of-ℓ* selections and assumed Paillier encryption, however any trapdoor



Client j
Bulletin Board
Mining Servers
$E[m_{j,i}]$
(+ proof of validity)
$E[m_{1,i}] \otimes \ldots \otimes E[m_{c,i}] =$
$E[m_{1,i} \oplus \ldots \oplus m_{c,i}] = T$
( ): can be omitted in semi-honest model
T
(+ proof of correctness)

Fig. 4. Sketch of a privacy-preserving frequency miner.

discrete logarithm with the homomorphic encryption property can be used instead (please refer to [17] for a list of candidate cryptosystems).

6.4. Security considerations

The above approach, if instantiated with Paillier encryption provides computational privacy under the *Decisional Composite Residuosity Assumption* (DCRA), which is required to show that a Paillier encryption is semantically secure [17]. If non-interactive proofs of validity are employed, the same result holds under the *random oracle* model. For verifiability, a client is able to check whether its encrypted input is valid and has been taken into account by the mining system. Each client is also able to compute the homomorphic product of the elements on the board and thus verify the accuracy of the mining process. In the presence of malicious or faulty miners, robustness is a result of the setup for a threshold cryptosystem. The reader may also refer to [17] for formal arguments concerning the threshold version of the Paillier cryptosystem and the ZK proofs.

6.5. Efficiency considerations

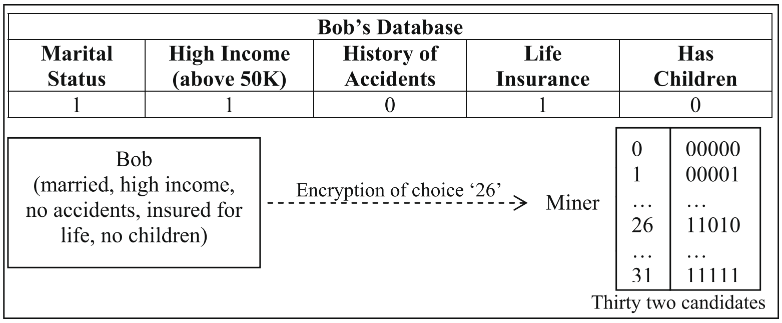
In the (simplest) semi-honest case, where ZK proofs and threshold decryption are not considered, each client performs two public-key operations, one for encrypting and one for digital signing the selection before uploading to the board. This is considered affordable, even for a client with limited computer power. For verifiability, correctness of the tally is also confirmed using $O(c)$ multiplications. The communication complexity is linear in the size of the Paillier modulus and the number c of clients. A message consists of one ciphertext and one signature accompanied with a digital certificate. Assuming that the size of N is L bits, then the size of a Paillier ciphertext is $2L$ bits and the communication cost for each user is $4L$ bits, for a similar signature modulus. The computation cost for the miner(s) is $O(c)$ multiplications and one decryption operation. Compared to the scheme in [1], which is limited to *1-out-of-2* selections, our approach has more efficient key generation and tallying process, while the client computation and total communication costs are similar.

For a full scheme with practical complexity, even in the malicious threat model, we propose the adoption of the generalized Paillier encryption as proposed in [18]. Compared with [17], efficiency is improved as the ballot size is reduced to $O(L \cdot log\ell)$, while the work for computing the final tally is linear in ℓ. They also proposed a threshold variant of the generalized system. The reader may refer to [18] for further details and formal arguments.

6.6. Mining with k-out-of-ℓ protocols

Protocols for *1-out-of-*ℓ selections can easily be adapted to support *k-out-of-*ℓ selections, where $k \leqslant \ell$. An easy generalization, with some loss of performance, would be to send k encrypted messages [18]. Another trivial solution is to encode all possible ℓ-bit numbers as separate candidates, thus producing a set of 2^{ℓ} candidates. Fig. 5 depicts the trivial approach in the *C2S* setting, where the problem of allowing *k-out-of-*ℓ selections from one record with ℓ features is reduced to a *1-out-of-*2^{ℓ} multi-candidate protocol.

Selections of *k-out-of-*ℓ elements could also be useful in a *S2S* setting, where the database is horizontally partitioned into a set of c client partitions, with each client possessing R full records of transactions. In this case, Bob would send up to R encrypted messages to the miner, where R is equal to the rows of the table in his partition. In the semi-honest model however, this $O(L \cdot R)$ size could be further reduced to $O(L)$, by using the following "trick": each client also acts as a miner, performs $O(R)$ modular additions and computes a local result as $T_{local} = \sum_{u=1}^{R} M_u$ where M_u is the selection for the uth row of the database. Then, the client encrypts T_{local} with the public key of the system miner and submits it to the bulletin board. Intuitively, the above trivial approaches are as secure as the *1-out-of-*ℓ scheme, however this assertion needs to be supported by future work.



Bob's Database				
Marital Status	High Income (above 50K)	History of Accidents	Life Insurance	Has Children
1	1	0	1	0

Bob
(married, high income, no accidents, insured for life, no children)

0	00000
1	00001
…	…
26	11010
…	…
31	11111

Fig. 5. A trivial way to turn a *k-out-of-*ℓ scheme into a *1-out-of-*ℓ scheme.

7. Case: PPDM in Random Forests (RF) classification

7.1. Introducing standalone RF

Throughout recent years, numerous attempts in presenting ensemble of classifiers have been introduced. The main motivation that underlies this trend is the need for higher prediction accuracy of the task at hand. *Random Forests* [56] are a combination of tree classifiers such that each tree depends on the value of a random vector, sampled independently and with the same distribution for all trees in the forest. The generalization error depends on the strength of the individual trees in the forest and the correlation between them. Using random selection of features to split each node yields error rates that compare favorably to Adaboost [57], and are more robust with respect to noise. While traditional tree algorithms spend a lot of time choosing how to split at a node, Random Forests put very little effort into this.

A random forest multi-way classifier $\Theta(x)$ consists of a number of trees, where each tree is grown using some form of randomization. The leaf nodes are labelled by estimates of the posterior distribution over data class labels. Each internal node contains a test that best splits the space of data to be classified. A new, unseen instance is classified by being inserted to every tree and by aggregating the reached leaf distributions. The process is depicted in Fig. 6. Randomness can be injected at two points during training: either in sub-sampling the training data so that each tree is grown using a different subset or in selecting the node tests. In our approach, we shall discuss the former situation, and argue that by using privacy-preserving protocols in randomly selected instances we support the creation of robust RF, thus allowing for effective mining in horizontally-partitioned data sets.

7.2. Privacy-preserving RF for horizontally-partitioned (HP) data

By the term horizontally-partitioned data, we mean that parties ($\geqslant 3$) collect values from the same set of features but for different objects. Their goal is to find an improved function for predicting class values of future instances, yet without sharing their data among each other. Thus, we enroll a collaborative approach where data need to be shared in a secure manner, and the final model will predict class labels without knowing the origin of the test instance. Similar to previous approaches such as [27], classification is performed individually, on each party's site, but the main contribution on the field is that during training, data from other parties are used in order to enhance randomness, thus increase the obtained classification accuracy. However, an assumption needs to be taken into account: data are sharing a common distribution. For example, suppose we have three different bank institutions, sharing financial information on their customers in a HP manner (e.g., they all use features such as *age, occupation, income, marital status* and *sex*). In order to have a robust RF classifier, data has to follow a similar distribution among banks, meaning that if one bank owns data on a specific group of customers (e.g., professors) and the others own data about a totally different group (e.g., farmers), the obtained accuracy would be severely deteriorated. We exploit the two strengths of RF i.e., randomness and voting. The former deals with the issue of premature termination of the tree learning process while the latter confronts data sparseness problems in an effective manner. In this work, we consider a protocol that allows for injecting randomness into trees during learning and allow voting over the majority class among all parties at classification time. More specifically, we shall discuss Random Input Forests (RI) learning from HP data sets and utilizing the forest structure to classify previous unseen test instances, originating from one of the distributed database parties.

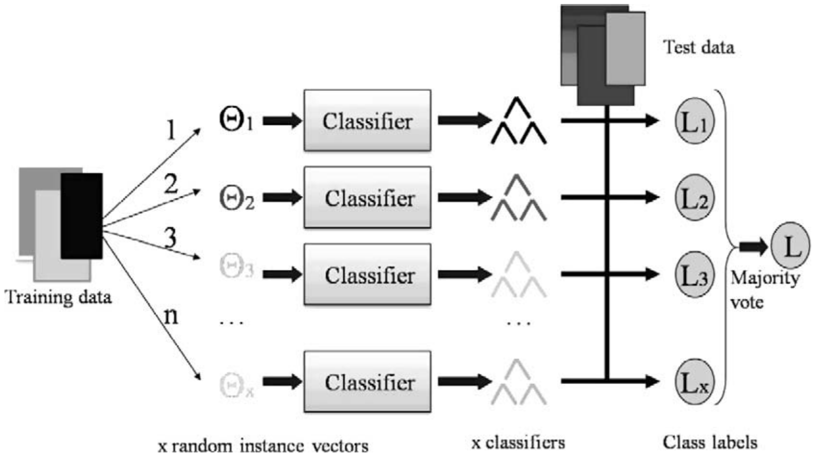


Fig. 6. Hierarchical decomposition of an RF classifier on a non-distributed data set.

7.3. Random input forests

Our privacy-preserving approach, used for training Random Forests at each party's side by inserting randomness from the other ones, is consisted of two distinct phases. In the former one, each party is collaborating using the procedure proposed by [27], in order to collect the whole set of available values per each attribute. This knowledge is particularly important for the next phase, where each party will require a certain number of instances from the others (again, we note that more than three parties are needed). The complete algorithm is as follows:

Each party selects K trees to grow:

- Build each tree by:
  - Selecting, at random, at each node a small set of features (F) to split on (given M features). From related research, common values of F are:
    1. $F = 1$
    2. $F = \log_2(M) + 1$

       F is held constant while growing the forest. Create a random instance based on the values of the complete feature set and ask the other parties to vote if they own it. Since F is significantly smaller that M, the number of candidate instances that each party will create is computationally efficient.
  - For each node split on the best of this subset (using oob instances),
  - Grow tree to full length. There is no pruning.

To classify a new, unseen instance X, collect votes from every tree of the forest of each party and use a general majority voting scheme to decide on the final class label.

Note that in case parties have agreed to send their data to a trusted site, e.g., a *Data Miner*, the complexity of the RI forest algorithm mentioned before is significantly reduced. More specifically, instead of each party generating random instances and ask other parties to vote in order to reveal if someone else owns this example, the Data Miner asks all parties to submit votes on their data set. By using the PPDM approach explained in Section 6 the Data Miner can obtain the number of each instance co-occurrence within the whole data set. Therefore, at learning time, each party initially uses its own data and then asks the Data Miner to reveal which K other instances have frequency larger than a given threshold s within the total data set. As depicted in the next section, this approach allows for building more robust classifiers based on RF, yet using PPDM protocols in order not to reveal which party owns each data instance.

The advantage of such a classifier is based on the improved manner of inserting randomness to the core classification trees during training time, yet protecting private information of data records. Since recent trends in classification systems deal with the use of ensemble methodologies, we argue in favor of a novel distributed algorithm that makes use of randomness in order to improve the internal prediction model. Potential applications that could benefit from such an implementation include financial institutions, insurance vendors, medical institutions and sectors of government infrastructures.

7.4. Experimental evaluation of PPDM using RF

We implemented our privacy-preserving Random Forests algorithm in Java, by augmenting an existing implementation of the Paillier homomorphic cryptosystem,[1] in order to match with the variation described in Section 6.6. Furthermore, we used the WEKA machine learning algorithm platform for generating two different random binary data sets, consisting of 10 and 20 binary attributes respectively with 1000 data instances each. Note that this will be referred to as *Total Set*. We ran our experiments on a Dual Core 2 GHz PC with 2GB of memory under Netbeans. As for the RF builder, we incorporated the WEKA library that deals with RF tree classifiers within our project. In our experiments, the length of the modulus for Paillier encryption was 512 bits. We randomly assigned generated data to three different parties (i.e., the *Partial Set*) without posing strict HP data set restrictions (i.e., an instance could belong to more than one party). The key-generation time (only performed once) was 1.5 s. As mentioned in the RI Forest protocol, each party encrypted its Partial Set and sent in to the Data Miner. The Data Miner received all encryptions and by performing the operations described in Section 6.3, obtained the number of votes that reveal how many times a given instance is found within the total data set. The former operation has computational time less than 1 ms for each instance while the Data Miner's computations are somewhat longer but still quite efficient (they are deteriorating as the number of parties and the size of the Total Set increases). For example, for three parties and 1000 examples within the Total Set, it took approximately 2.1 s to learn the counts of each instance. Nevertheless, these values can be pre-computed off-line, prior to the RI forests learning phase. We empirically set each party to ask for 70 more instances from the Miner, having a threshold frequency of 1 (i.e., each party adds an instance if it is owned by at least one other party). Upon creation of the new, enhanced data set, each party classified its Original Partial Set, using 10-fold cross validation and also classified its new, augmented Partial Set in order to evaluate the performance of RF when randomness is injected. Fig. 7 portrays the classification performance in terms of F-measure (which is the harmonic mean of precision/recall metrics) for party A, on the two generated data sets, using the initial Partial Set and the augmented data set from the aforementioned privacy-preserving protocol.

[1] http://www.csee.umbc.edu/~kunliu1/research/Paillier.java.

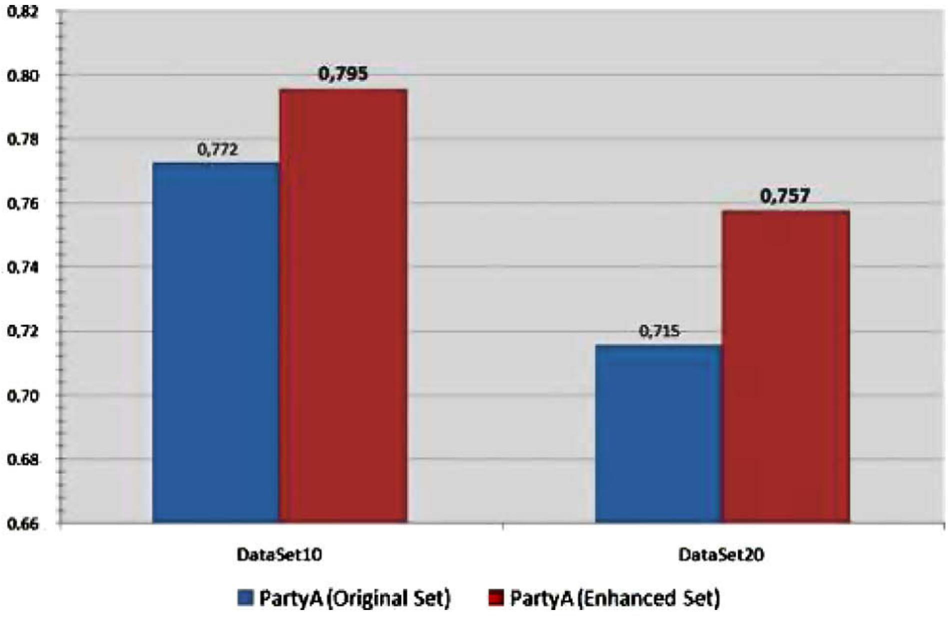


Fig. 7. Classification using RF for Party A in both data sets, using two types of tree randomness.

Results are similar for the other two parties as well. As we could observe, by introducing new, previously unseen examples, more robust RF are built, able to better classify data, by a varying factor of *2–4.5%*.

8. Future work

We expect that research in efficient cryptography for large-scale PPDM will be continued towards solutions that balance the trade-off between efficiency and security. Future work will also show how cryptography-based approaches for PPDM can be combined with techniques for controlling access to individual entries in statistical databases, in order to further improve the security of the mining process. For future work, we intend to extend our simple scheme in order to build the confidence and support metrics on a distributed system for mining association rules, where the transaction database is fully distributed among the clients of the system. In addition, we intend to further implement the training methodologies and applying them to real-world, large-scale databases for evaluation. We are particularly motivated by recent trends towards multi-processor systems and parallelization, that could significantly improve the performance of the system.

References

[1] Z. Yang, S. Zhong, R.N. Wright, Privacy-preserving classification of customer data without loss of accuracy, in: SDM' 2005 SIAM International Conference on Data Mining, 2005.
[2] M.-S. Chen, J. Han, P.S. Yu, Data mining: an overview from a database perspective, IEEE Transactions on Knowledge and Data Engineering 08 (6) (1996) 866–883.
[3] Y. Lindell, B. Pinkas, Privacy preserving data mining, in: CRYPTO'00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology, Springer-Verlag, London, UK, 2000, pp. 36–54.
[4] C. Clifton, D. Marks, Security and privacy implications of data mining, in: Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD'96), 1996.
[5] N. Zhang, S. Wang, W. Zhao, A new scheme on privacy-preserving data classification, in: KDD'05: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, ACM, New York, NY, USA, 2005, pp. 374–383.
[6] A. Prodromidis, P. Chan, S.J. Stolfo, Meta-learning in distributed data mining systems: issues and approaches, Advances in Distributed and Parallel Knowledge Discovery (2000) 81–114.
[7] J. Vaidya, C. Clifton, Privacy preserving association rule mining in vertically partitioned data, in: KDD'02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2002, pp. 639–644.
[8] M. Kantarcoglu, J. Vaidya, Privacy preserving naive Bayes classifier for horizontally partitioned data, in: IEEE ICDM Workshop on Privacy Preserving Data Mining, Melbourne, FL, 2003, pp. 3–9.
[9] J. Vaidya, H. Yu, X. Jiang, Privacy-preserving svm classification, Knowledge and Information Systems 14 (2) (2008) 161–178.
[10] J. Zhan, Privacy-preserving collaborative data mining, Computational Intelligence Magazine 3 (2) (2008) 31–41.
[11] W. Jiang, C. Clifton, M. Kantarcioglu, Transforming semi-honest protocols to ensure accountability, Data and Knowledge Engineering 65 (1) (2008) 57–74.
[12] N.R. Adam, J.C. Wortmann, Security-control methods for statistical databases: a comparative study, ACM Computing Surveys 21 (4) (1989) 515–556.
[13] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin, Y. Theodoridis, State-of-the-art in privacy preserving data mining, SIGMOD Record 33 (1) (2004) 50–57.

[14] U. Maurer, The role of cryptography in database security, in: SIGMOD'04: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY, USA, 2004, pp. 5–10.

[15] O. Goldreich, S. Micali, A. Wigderson, How to play any mental game, in: STOC'87: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, ACM, New York, NY, USA, 1987, pp. 218–229.

[16] R. Cramer, R. Gennaro, B. Schoenmakers, A secure and optimally efficient multi-authority election scheme, European Transactions on Telecommunications 8 (5) (1997) 481–490.

[17] O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, G. Poupard, Practical multi-candidate election system, in: PODC'01: Proceedings of the 20th Annual ACM Symposium on Principles of Distributed Computing, New York, NY, USA, ACM, 2001, pp. 274–283.

[18] I. Damgard, M. Jurik, J. Nielsen, A generalization of Paillier's public-key system with applications to electronic voting, 2003.

[19] E. Magkos, M. Maragoudakis, V. Chrissikopoulos, S. Gritzalis, Accuracy in privacy-preserving data mining using the paradigm of cryptographic elections, in: PSD'08: Privacy in Statistical Databases, Lecture Notes in Computer Science, vol. 5262/2008, Springer, 2008, pp. 284–297.

[20] R. Agrawal, R. Srikant, Privacy-preserving data mining, in: Proceedings of the ACM SIGMOD Conference on Management of Data, ACM Press, 2000, pp. 439–450.

[21] A. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke, Privacy preserving mining of association rules, in: KDD'02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2002, pp. 217–228.

[22] S.J. Rizvi, J.R. Haritsa, Maintaining data privacy in association rule mining, in: VLDB'02: Proceedings of the 28th International Conference on Very Large Data Bases, VLDB Endowment, 2002, pp. 682–693.

[23] D. Agrawal, C.C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: PODS'01: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, New York, NY, USA, 2001, pp. 247–255.

[24] I. Dinur, K. Nissim, Revealing information while preserving privacy, in: PODS'03: Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, New York, NY, USA, 2003, pp. 202–210.

[25] C.G.K. Liu, H. Kargupta, A survey of attack techniques on privacy-preserving data perturbation methods, in: C. Aggarwal, P. Yu (Eds.), Privacy-Preserving Data Mining: Models and Algorithms, Springer-Verlag, 2008.

[26] A.C.-C. Yao, How to generate and exchange secrets (extended abstract), in: FOCS, 1986, pp. 162–167.

[27] M. Kantarcioglu, C. Clifton, Privacy-preserving distributed mining of association rules on horizontally partitioned data, IEEE Transanctions on Knowledge and Data Engineering 16 (9) (2004) 1026–1037.

[28] R. Wright, Z. Yang, Privacy-preserving Bayesian network structure computation on distributed heterogeneous data, in: KDD'04: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, ACM, New York, NY, USA, 2004, pp. 713–718.

[29] J. Zhan, S. Matwin, L. Chang, Privacy-preserving collaborative association rule mining, Journal of Network and Computer Applications 30 (3) (2007) 1216–1227.

[30] S. Goldwasser, Multi party computations: past and present, in: PODC'97: Proceedings of the 16th Annual ACM Symposium on Principles of Distributed Computing, ACM, New York, NY, USA, 1997, pp. 1–6.

[31] B. Pinkas, Cryptographic techniques for privacy-preserving data mining, SIGKDD Explorations Newsletter 4 (2) (2002) 12–19.

[32] W. Du, Z. Zhan, Building decision tree classifier on private data, in: CRPIT'14: Proceedings of the IEEE International Conference on Privacy, Security and Data Mining, Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 2002, pp. 1–8.

[33] Z. Yang, S. Zhong, R.N. Wright, Towards privacy-preserving model selection, in: PinKDD'07: Privacy, Security, and Trust in KDD, vol. 4890/2008.

[34] S. Zhong, Privacy-preserving algorithms for distributed mining of frequent itemsets, Information Sciences 177 (2) (2007) 490–503.

[35] M. Morgenstern, Security and inference in multilevel database and knowledge-base systems, SIGMOD Record 16 (3) (1987) 357–373.

[36] J. Domingo-Ferrer (Ed.), Inference Control in Statistical Databases, From Theory to Practice, Lecture Notes in Computer Science, vol. 2316, Springer, 2002.

[37] D. Woodruff, J. Staddon, Private inference control, in: CCS'04: Proceedings of the 11th ACM Conference on Computer and Communications Security, ACM, New York, NY, USA, 2004, pp. 188–197.

[38] G. Jagannathan, R.N. Wright, Private inference control for aggregate database queries, in: ICDMW'07: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops, IEEE Computer Society, Washington, DC, USA, 2007, pp. 711–716.

[39] P. Samarati, L. Sweeney, Generalizing data to provide anonymity when disclosing information (abstract), in: PODS'98: Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, ACM, New York, NY, USA, 1998, p. 188.

[40] H. Kargupta, K. Das, K. Liu, A game theoretic approach toward multi-party privacy-preserving distributed data mining, in: Eleventh European Conference on Principles and Practice of KDD (PKDD), Warsaw, Polland, 2007.

[41] J. Benaloh, D. Tuinstra, Receipt-free secret-ballot elections (extended abstract), in: STOC'94: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, ACM, 1994, pp. 544–553.

[42] M. Hirt, K. Sako, Efficient receipt-free voting based on homomorphic encryption, in: Proceedings of the Advances in Cryptology – EUROCRYPT'00, LNCS, vol. 1807, Springer-Verlag, 2000, pp. 539–556.

[43] Y.G. Desmedt, Y. Frankel, Threshold cryptosystems, in: CRYPTO'89: Proceedings on Advances in Cryptology, Springer-Verlag New York, Inc., New York, NY, USA, 1989, pp. 307–315.

[44] O. Goldreich, S. Micali, A. Wigderson, Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems, Journal of the ACM 38 (3) (1991) 690–728.

[45] A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, in: Proceedings on Advances in Cryptology – CRYPTO'86, Springer-Verlag, 1987, pp. 186–194.

[46] J.D.C. Benaloh, Verifiable secret-ballot elections, Ph.D. thesis, New Haven, CT, USA, 1987.

[47] M. Reiter, The rampart toolkit for building high-integrity services, in: Proceedings of the Interational Conference on Theory and Practice in Distributed Systems, LNCS, vol. 938, Springer, 1995, pp. 99–110.

[48] D. Gritzalis (Ed.), Secure Electronic Voting: Trends and Perspectives, Capabilities and Limitations, Kluwer Academic Publishers, 2003.

[49] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory IT-22 (6) (1976) 644–654.

[50] R.J. Cramer, M. Franklin, L.A. Schoenmakers, M. Yung, Multi-authority secret-ballot elections with linear work Tech. rep., Amsterdam, The Netherlands, 1995.

[51] B. Schoenmakers, A simple publicly verifiable secret sharing scheme and its application to electronic voting, Lecture Notes in Computer Science 1666 (1999) 148–164.

[52] T.E. Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in: Proceedings of CRYPTO 84 on Advances in Cryptology, Springer-Verlag, 1985, pp. 10–18.

[53] E. Magkos, P. Kotzanikolaou, C. Douligeris, Towards secure online elections: models, primitives and open issues, Electronic Government International Journal 4 (3) (2007) 249–268.

[54] A. Fujioka, T. Okamoto, K. Ohta, A practical secret voting scheme for large scale elections, in: Proceedings of the Advances in Cryptology – AUSCRYPT'92, LNCS, vol. 718, Springer-Verlag, 1992, pp. 244–251.

[55] P. Paillier, Public-key cryptosystems based on discrete logarithms residues, in: Eurocrypt 99, LNCS, vol. 1592, Springer-Verlag, 1999, pp. 221–236.

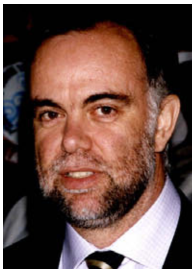
[56] L. Breiman, Random forests, Machine Learning Journal 45 (1) (2001) 32–73.

[57] L. Breiman, Bagging predictors, Machine Learning Journal 26 (2) (1996) 123–140.

Emmanouil Magkos was born in Athens, in 1975. He received his first degree in Computer Science from the University of Piraeus, Greece in 1997. In April 2003 he received a Ph.D. in Information Security and Cryptography from the University of Piraeus, Greece. Since 2007 he is affiliated with the Department of Informatics of the Ionian University, in Corfu, Greece, where he holds the position of Lecturer in Computer Security and Cryptography. He has (co-) authored more than 30 papers in international journals and conferences related to Information Security and Cryptography. He also acts as a reviewer for several international journals and conferences. His current research interests include Privacy and Key Management in wireless ad-hoc networks, monitoring infrastructures for Malware Propagation, Privacy Preserving Data Mining Systems, Integrity in Secure Logging systems, Cryptographic Security of Internet Voting systems.

Manolis Maragoudakis was born in Athens. He obtained his PhD in Artificial Intelligence from the Department of Electrical and Computer Engineering, University of Patras. He is a computer scientist, graduated from the Department of Computer Science, University of Crete. The title of the PhD dissertation was: "Modeling and reasoning under uncertainty in dialogue and other natural language systems using Bayesian networks". He is currently a lecturer at the Department of Information and Communication Systems Engineering, University of the Aegean, in the research field of "Data Mining". He is also a reviewer at the "IEEE Transactions on Knowledge and Data Engineering" and "Artificial Intelligence Tools" journal. He has previously taught at the Ionian University, Greece as an assistant professor. He has published in more than 50 articles in the field of Artificial Intelligence and Data Mining in international journal, conferences and workshops. He is a Member of the IEEE and the Hellenic Artificial Intelligence Society.

Vassilis Chrissikopoulos is Professor of informatics in the Department of Informatics at the Ionian University. He received his BSc from the University of Thessaloniki, Greece (1976), and his M.Sc and PhD from University of London (1979, 1983). During the period 1985–2000 he was member of staff (Ass. Prof., Assoc. Prof., and Professor) in the Department of Informatics at the University of Piraeus. During the period 2000–2007 he was affiliated as a Professor in the Department of Archives and Library Science at the Ionian University. His research interests include Information security and cryptography, e_commerce, mobile agents, e_voting and digital libraries. Prof. Chrissikopoulos has participated in several research projects as a coordinator or as a member, funded by Greece or European Community. He also was member of several technical committees and working groups on subject relates to informatics and information security. He is a member of the Greek Mathematical Society, and of the Greek Computer Society.



Stefanos Gritzalis holds a BSc in Physics, an MSc in Electronic Automation, and a PhD in Informatics all from the University of Athens, Greece. Currently he is the Head of the Department of Information and Communication Systems Engineering, University of the Aegean, Greece and the Director of the Laboratory of Information and Communication Systems Security (Info-Sec-Lab). He has been involved in several national and EU funded R&D projects in the areas of Information and Communication Systems Security. His published scientific work includes several books on Information and Communication Technologies topics, and more than 180 journal and national and international conference papers. The focus of these publications is on Information and Communications Security and Privacy. He has leaded more than 25 international conferences and workshops as General Chair or Program Committee Chair, and has served on more than 150 Program Committees of international conferences and workshops. He is an Editor-in-Chief for 1 journal, an Editorial Advisory Board member for more than 10 journals and a Reviewer for more than 35 journals. He was an elected Member of the Board (Secretary General, Treasurer) of the Greek Computer Society. He is a Member of the ACM, and the IEEE.