Privacy-Preserving Information Markets for Computing Statistical Data

Aggelos Kiayias[⋆], Bülent Yener[⋆⋆], and Moti Yung

[1] Computer Science and Engineering, University of Connecticut
Storrs, CT, USA. aggelos@cse.uconn.edu
[2] Computer Science Department,
RPI, Troy, NY, USA.
yener@cs.rpi.edu
[3] Google Inc. and Computer Science, Columbia University
New York, NY, USA. moti@cs.columbia.edu

Abstract. Consider an "information market" where private and potentially sensitive data are collected, treated as commodity and processed into aggregated information with commercial value. Access and processing privileges of such data can be specified by enforceable "service contracts" and different contract rules can be associated with different data fields.

Clearly the sources of such data, which may include companies, organizations and individuals, must be protected against loss of privacy and confidentiality. However, mechanisms for ensuring privacy per data source or data field do not scale well due to state information that needs to be maintained. We propose a scalable approach to this problem which assures data sources that the information will only be revealed as an aggregate or as part of a large set (akin of k-anonymity constraints).

In particular, this work presents a model and protocols for implementing "privacy preserving data markets" in which privacy relies on the distribution of the processing servers and the compliance of some (a quorum) of them with the service contract. We then show how to compute statistical information important in financial and commercial information systems, while keeping individual values private (e.g., revealing only statistics that is performed on a large enough sample size). In detail, we present two novel efficient protocols for privacy-preserving S-moments computation (for $S = 1, 2, \ldots$) and for computing the Pearson correlation coefficients.

1 Introduction

Internet users today are often requested to pass personal information to their health-care providers, to their banks, to insurance companies and other service providers. Similarly, organizations have to disclose private individual data to

[⋆] Research partly supported by NSF SGER Award 0751095 as well as partly by NSF Career Award 0447808.

[⋆⋆] Research partly supported by NSF SGER Award 0751069.

suppliers and contractors in order to satisfy supply chain requirements. In fact, such information that may be collected by a primary service provider can be sensitive and may be protected by privacy disclosure acts (e.g., HIPAA is such act in the United States) or by business confidentiality agreements. In many cases, appropriately processed data are valuable market assets since they can be used to improve services, increase sales, etc. Therefore it is often important to transfer individual data items from the primary market where they are collected to a "secondary market" where other parties will further process them (e.g., will compute statistics of important parameters) and potentially disclose these secondary processed outcomes as opposed to the original (much more private) data.

We note that the privacy implications in these "data market" settings are dire since the users that provide data have no control on how primary service providers outsource their data. In addition, there is no way to enforce privacy and it is also possible that secondary market entities reside outside the jurisdiction where the data were collected originally (so even a legal procedure can be complicated). Furthermore, collected outsourced data may be stored in data-warehouses such as LexisNexis, can be sold to other parties for data mining, or in the worst case they can be exposed to unauthorized malicious entities who, exploiting a security vulnerability, may access this sensitive information. While these scenarios raise serious privacy concerns, it should be stressed again that there is a clear need for knowledge discovery in the secondary market: First for commercial (e.g., marketing, pricing, improving market efficiencies, service assessment and billing, revising insurance policy estimations), and secondly, for research purposes; and even for safety reasons (e.g., identifying public health hazards, realizing outbreaks such as epidemics or biological warfare instances, market research, etc.). Still in the present situation, "data producers" (i.e., users like all of us, and organizations) have little control over who, how, and what exactly is done with private and sensitive data that are communicated to a secondary market.

The current uncertainty of the way that private information may be taken advantage of, brings forth another important concern: users increasingly use falsification of their personal information when they are filling out Internet forms. Indeed, a number of recent reports [31, 44, 4, 11] show that somewhere from 20% to 50% of online users have provided false data when confronted with an online form with the aim of protecting their privacy. The amount of false information that is collected reduces the usefulness of information databases for legitimate purposes and leads to a waste of resources.

1.1 Our contributions

Market Trust Infrastructure: We claim that what is needed, given the current situation, is a *trust infrastructure* that will assure users that their personal data is not revealed and that collection is secured at the primary service provider. Further assurance involves the fact that the secondary market aggregation and mining processing has security, integrity and validation built into it. (Note that

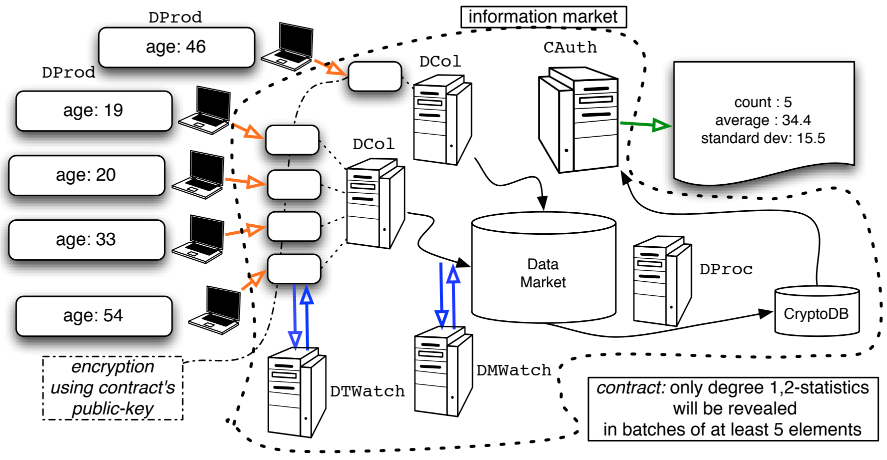

Fig. 1. Privacy Preserving Information Processing. Data producers `DProd` contribute scrambled private information to data collectors `DCol`; scrambled data enter the data-market where they are freely marketed protected by their cryptographic contract. The parties `DTWatch` and `DMWatch` are optional and have the role to ensure that data collected from `DCol` are indeed corresponding to real users. Eventually data are removed by the data market and packed into a crypto-database by the data processor `DProc`. The contract authority `CAuth` verifies the properties of the associated contract and engages in a protocol with `DProc` that reveals the required outcome of the processing. `DProc` should be able to preprocess the cryptodatabase to reduce the computational cost on `CAuth` and the communication complexity.

currently, certificate authority infrastructure is a trust infrastructure for users' credentials but there is no similar entity for information markets).

Privacy Preserving Information Processing. The present work puts forth the notion of "Privacy-Preserving Information Processing" (PIP) to deal with the above basic problem. Central to our scenario is the *privacy-contract*: an agreement between the user and the service provider that will enable the generation of a special record. This is a type of an electronic "smart contract" as the ones advocated in [52]. The framework provides mechanisms for limiting data exposure and manipulation according to the contract, it also provides methods for validating compliance under the contract, in an analogous way to digital signature validation inside a PKI. More specifically: in PIP the data collection operation extends the protocol between the user and the primary service provider where the user (based on local privacy settings) furnishes to the provider, in addition to any other necessary information for primary service, also a contract-enforced-record (CER) which defines rules regarding the encrypted information in the record. In some settings the CER may contain rules contributed by both the service provider and the user (for example the case where a patient interacts with a primary care physician).

To enable secondary data markets that are under control, each CER is based on a template form and contains a sequence of fields that are encrypted according to specialized encryption functions (to be detailed later on). The primary service provider, in turn, has the choice to outsource CERs to a secondary market where they can be processed by following a data processing protocol that will involve the "contract-authority," a distributed entity that is used to safeguard privacy. The contract-authority is implemented in a distributed fashion by various entities possibly including (some of) the users themselves and service providers that wish to be trustees in the privacy-preserving operations. Based on threshold cryptography techniques, access to result decryption will be enforced by quorum control.

Then, PIP is a natural extension of the PKI concept where the data-collector offers a certificate to the data-producer. The certificate in this case has a much broader scope though: not only it provides the authentication of the server identity, but it also includes the following information: (i) the data-structure type of the data that the data-collector is soliciting, (ii) the contract that describes the purpose of the data-collection and its conditions, and possibly (iii) a cryptographic engine that enhances the client's machine with encryption capabilities. All the above information is signed by a *contract authority* (in the same way that the certification authority would sign a server's public-key).

The data producer, after verifying the certificate, supplies its data and is ensured (by the contract authority) that the data it provides will not be used in contract violation. This cryptographic contract binding is achieved by encrypting the data using the included cryptographic engine. *We stress that this means that the data will not be available to the secondary market in cleartext form.* Note that the data-collector will still "own" the submitted data but these will only be identified by descriptive fields (tags) such as "name", "age", "income" while the respective values would be enciphered. At this stage the data-collecting server may store the data or even *trade them freely* as a commodity in a data-market.

It should be stressed at this point that in PIP, nothing changes from the point of view of the user/data-producer: the user software still verifies a certificate (as in a SSL/TLS handshake) and then prompts the human operator to enter the data in an online form that may also be complemented by data produced by the primary service provider (e.g., the user's cat-scan in a medical application).

Naturally, the data also needs to be processed. This requires that the data elements are passed through the authorization of the contract authority who verifies that the submission is compliant with the stated contractual agreements. In particular, data is assembled into a "cryptodatabase", potentially get pre-processed by a data processing entity and is submitted to the contract authority, that verifies the processing request and produces the appropriate aspect of the data processing as described in the contract, to the requesting entity.

PIP Operation	Encoding Size	Communication	Computation
up to S-statistics	$\log N + S \cdot len$	$S \cdot (\log N + S \cdot len)$	$\mathtt{dec}(\nu)$
correlation coefficient	$\log N + 2 \cdot len$	$4(\log N + 2 \cdot len)$	$< 4 \cdot \sqrt{\#\mathsf{D}_1 \cdot \#\mathsf{D}_2} \cdot \mathtt{dec}(\nu)$

Fig. 2. Summary of our results. S is a parameter that specifies the highest moment that is required to be computed; ν is a cryptographic security parameter assumed to satisfy $\nu = \Omega(\log N)$ where N is the sample size (i.e., the number of data producers). Data producers are assumed to draw their values from a space D in the first system and from $\mathsf{D}_1, \mathsf{D}_2$ for the correlation system; *len* is the maximum size required to encode any of the numerical elements in $\mathsf{D}, \mathsf{D}_1, \mathsf{D}_2$; $\mathtt{dec}(\nu)$ is the time required to decrypt a ciphertext corresponding to security parameter ν.

Constructing PIP *schemes.* Next, we identify the major challenge in designing a PIP system which is the following: For a given processing operation, the challenge is to design a *cryptodatabase processing* operation (that is accompanied by an appropriate encryption scheme) so that the contract authority communication and computation complexity becomes a small function of the size of the output of the processing operation (which is the natural lower bound for the complexity for performing the computation over the cleartext data — just from the need to produce all the output).

Based on this, we concentrate on statistics for information processing that is crucial in the context of collecting and processing financial numerical data. Using novel cryptographic constructions together with recently devised existing encryption systems, we present two near optimal PIP systems. These systems enable the evaluation of statistical information from collected numerical data in a private fashion. In particular, we consider the setting where data producers contribute numerical data (e.g., their income, age, revenue, profit etc.) drawn from domains $\mathsf{D}_1, \mathsf{D}_2, \ldots$ and the data processor wishes to extract the following statistical information:

- S-statistics and S-th moments for $S = 1, \ldots,$ that disclose the mean, standard deviation and higher moments (note that with more moments available better approximations of the sample distribution can be made available).
- the correlation coefficient between two samples that enables us to relate two distributions via their corresponding samples.

In particular, if data producers provide the values $v_1, \ldots, v_N$, the data processor can use our first PIP system to extract the r-th sample central moment for $r = 1, \ldots, S$, or more generally can approximate the sample statistical distribution up to the S-th cumulant. Recall that computing cumulants enables one to approximate the Maclaurin expansion of the logarithm of the probability density function of the underlying population distribution. In our second PIP system we show how the data processor can extract the Pearson correlation coefficient from two data columns $v_1, \ldots, v_N, v'_1, \ldots, v'_N$ submitted by the sample of N users.

Our results are summarized in figure 1.1. Note that the optimal measure is $S \cdot len$ for the first PIP system and len for the second; it follows that our first scheme almost matches the optimal measure (it is polylogarithmically related to

N and multiplied by the parameter S that specifies the highest moment to be computed — in general S assumed a small constant); our second PIP system has similarly favorable communication; on the other hand, the required computation is proportional to the size of the data space as opposed to proportional to the output of the processing operation which is len; still the size of the data space is always a tractable value in practice.

It should be stressed that our constructions offer absolute privacy (i.e., they reveal nothing but the required output) under the cryptographic conditions and the threshold implementation of the contract authority.

We note though that the inclusion of the same CER to various different PIP operations may result in privacy violations that are unanticipated. For example, a record can be entirely revealed if two different sample mean calculations are performed where the only difference is including and excluding that item — this is a typical problem in statistical database queries. Resolving this issue – to the degree it can be solved – goes beyond the scope of the present work. One possible approach is to restrict subsequent inclusions of a certain CER to a PIP unless it is used in the same context (i.e., with the same CERs that it appeared in the first operation). Such approach would require an ever growing state that keeps track of past PIP operations. Still there are possible alternatives to that end and the basic infrastructure developed herein is consistent with other approaches to ensuring database privacy that can be useful in this respect, in particular adding noise to private data, [17] or using "baits" to catch misbehaving data collectors [29].

1.2 Related Previous Work

We next review a few areas that are related to our notion and explain in what ways our market model is different.

Secure function evaluation. Looking at PIP from a theoretical viewpoint, one can identify it as an instance of a secure multi-party computation with private inputs, a cryptographic primitive that has been studied extensively in the literature, and in fact generic protocols have been constructed that allow arbitrary functionality, see [27]. These protocols are not practical as they require large communication and computation costs; as advocated in [28], it is important to pursue more efficient instantiations of such "secure multi-party computations" and the instantiations that will be described in this proposal characterize an efficient sub-class of such generic protocols. More efficient cryptographic protocol constructions have, in fact, been successful for example a prominent has been electronic voting cf. [14, 12, 48, 49, 15, 33, 30, 35, 8, 6, 39–41].

Cryptographic database processing for privacy preserving data mining. Knowledge discovery with privacy concerns in terms of Privacy Preserving Data Mining (PPDM) was investigated in the context of secure computation in [43, 34, 53, 2, 45, 1, 22, 42, 32, 54, 36, 37]. In this setting the focus is on merging or processing data from multiple private datasets that should not be mutually disclosed. The databases are typically owned and operated by entities that may pre-process them and share cryptographically scrambled versions of database aspects. The fundamental difference between this approach and the present work is that the private data are available in a cleartext form to the database owner while in our approach the database owner is not trusted. In other words the previous methods can be used to assist well-to-do data-collectors and processors to adhere to their privacy statements. Nevertheless this approach does little to protect against the problem we are tackling in this work, namely to provide a safeguard mechanism that ensures contract enforcement as well as notification and dispute control at the user level.

Encrypted access control and processing. Enforcing access control through cryptographic means has been utilized numerous times in secure system design (e.g., for file storage cf. [7, 26] for hierarchical access control cf. [3, 13, 50, 47]). The PIP setting goes beyond such access control since it focuses on data collection and processing, i.e., access to the data is not only restricted but also requires ciphertext-based processing that should be combined to an aggregation capability as data from many data-producers need to be pulled together prior to processing.

Multi-party communication. A PIP system requires multi-party communication and coordination which is a challenging problem in distributed environments. Dealing with failures and corruptions in multi-party communication systems is the context of Byzantine agreement protocols, a subject that is extensively studied in the literature, e.g. [18, 19, 25, 24]. Byzantine agreement procedures, although they allow multi-party procedures to succeed under typically a threshold assumption on the number of failing parties, they do not constitute a very efficient approach for basing communication in multi-party systems. The approach followed here bases all communication of a multi-party system over a Client-Server communication infrastructure. The potential of the Client-Server communication model for basing security in multi-party computations, has been investigated in [5]. In this work, we employ the client-server architecture as a mechanism for communication, computation assistance and increased trustworthiness (we do not deal with the underlying reliability issues since modern communication is quite reliable and furthermore it is a different layer).

Utilizing Client Interaction. Dealing with privacy in data collection, the encipherment of collected data was also considered in the context of data mining in [56, 55, 9]. For example in [56], k-anonymity was discussed in the context of a non-trusted database holder; in the suggested approach the data-producers help the database processors produce the k-anonymized version of the database with interaction involving cryptographic operations extending beyond the initial data submission. Compared to the PIP framework we propose here, this approach violates the principle that in a PIP protocol, data producers should not be required to be active in other stages of the system beyond the original data collection stage. Instead in PIP we opt for a logical separation between the roles of contract authorities and data producers where the latter are still given the opportunity to be contract authority shareholders if some of them wish to participate in the trust infrastructure. Moreover the focus of the present work is to provide near optimal solutions to numerical data statistical computation whereas previous work focused on more generic tasks, e.g., [9], that if applied to our setting they would result in protocols that lack privacy (as all numerical values will be revealed as opposed to the final outcome of the computation).

2 The PIP Framework

In PIP there are four basic roles: data-producers (users), data-collectors (primary service providers), data-processors (secondary market entities), and contract-authority servers (that comprise the trust infrastructure). We will refer to these entities as

$$\langle \mathtt{DProd}, \mathtt{DCol}, \mathtt{DProc}, \mathtt{CAserver} \rangle$$

The operation of a privacy-contract-based system will comprise the following four basic operations:
Trust Infrastructure Maintainance. This stage is executed by the "cloud" of `CAserver` entities. The operation requires two parameters: ν a cryptographic security parameter and ρ a fraction that determines the percentage of `CAserver` that need to agree for a certain processing operation to take place. The main task of the setup stage is to provide a set of cryptographic keys that will be used in the formation of the CERs. The operation of the system assumes that all `CAserver` entities may fail or shut-down arbitrarily; moreover, it assumes that at any given moment no bigger than ρ fraction of servers is corrupted by an adversary.

– *Create Initial Key.* An initial group of n `CAserver` setup a cryptographic key pk so that each each server receives a share sk_i of pk so that any $t = \lceil \rho \cdot n \rceil$ shares can be used to reconstruct the secret but any smaller number reveals no information about the secret in the computational sense.
– *Add* `CAserver`. This is a protocol between $t = \lceil \rho \cdot n \rceil$ existing servers and a new entity that wish to become a shareholder. It results in the generation of an independent share and the outcome of this operation should be indistinguishable compared to the shares obtained by the $n + 1$ servers (n existing plus the new one) should they have executed the initial key creation step.
– *Remove* `CAserver`. `CAserver` entities may arbitrarily shut off and stop participating in the operations of the contract authority server cloud. Depending on the communication model other servers may need to update routing tables.
– *Shares Calibration.* Given that the add/remove server operations modify the number of servers it will be the case that $|t/n - \rho| \geq \epsilon$ where ϵ is a deviation threshold that is a parameter of the system. In such case, a set of t servers execute a protocol that results in a corrected t' threshold equal to $\lceil \rho \cdot n \rceil$.

The above operations can be achieved by employing threshold cryptography techniques: creating the initial key will be based on Shamir's secret-sharing [51] as used in distributed key generation of e.g., [10], while the add-user and share calibration protocols can be based on the poly-to-sum and sum-to-poly protocols of dynamic proactive secret-sharing [20] for example. The communication model that is assumed here is a full-broadcast channel that can be simulated by the cloud of servers using byzantine-agreement in a fully adversarial setting [18, 19, 25, 24]; while in practical settings weaker protocols are still sufficient, say employing a client-server based bulletin-board system.

The public-key is bound to the type of operation and data type of CER records. It is certified by a certification authority and listed in a public directory where users can recover it if needed. For a given privacy-contract $\mathcal{C}$ pertaining to some data type and operation, we will denote by $\mathsf{pk}_{\mathcal{C}}$ the public-key of the contract, by $\mathsf{D}_{\mathcal{C}}$ the data type of the data collected and by and by $f_{\mathcal{C}} : (\mathsf{D}_{\mathcal{C}})^* \rightarrow R_{\mathcal{C}} \cup \{\perp\}$ the type of operation that will be applied after data-collection under the contract $\mathcal{C}$ (note that it may be a class of functions as well but for simplicity we just list a single function for now). Note that we allow $f_{\mathcal{C}}(\boldsymbol{x}) = \perp$, which is to be interpreted that performing the operation $f_{\mathcal{C}}$ on data input $\boldsymbol{x} \in (\mathsf{D}_{\mathcal{C}})^*$ would be in contract violation (this is not a catch-all as a contract violation may be triggered by other conditions as well).

User setup stage. Each user can obtain a signing key; this key is incorporated into the user's software and acts like an authorization token. The primary service provider (say health-care provider) can identify the user using such credential. Moreover, the signing key enables the user to sign CER records so that the following are satisfied:

– *Anonymity of Signatures.* Signatures produced by two distinct users are computationally indistinguishable for any observer, including an entity that corrupted the primary and secondary service providers as well as the contract authority entities.
– *Claiming of Signatures.* Each user can use its signing-key to execute a protocol that will "claim" a posted signature as produced by this user. No user can claim signatures that were not produced by it.

The above operations can be based on the notion of traceable signatures [38] (as they constitute a subset of the requirements put forth there).

Data-collection. In this setting primary service providers will be engaging in communication with the users that will furnish the CERs to the service provider. Each CER will be encrypted under the public-key of the negotiated privacy contract $pk_{\mathcal{C}}$ and signed using the signing-key of the user (note that the anonymity of the signature ensures that no information about the identity of the user is leaked from the signature). In more detail, `DCol` may present to users/data-producers a form that will facilitate the data-collection operation. The interface will be part of the transaction that the data-producer `DProd` and `DCol` are engaged in. The data-collection will include the following steps: first the data-collector and the data-producer will engage in a contract negotiation stage; in simple deployments this will be manual (and as simple as checking that the user read a privacy statement and it accepts it). Still it is possible to build a more elaborate negotiation stage where a client-side sub-system (e.g., a web-browser extension) will

negotiate the right contract type out of the ones that the DCol offers based on privacy settings that the user may have selected in advance. At the end of the negotiation stage the DProd will have verified that it is DCol that is collecting the data and that the data will be collected under the conditions of contract $\mathcal{C}$ that DProd accepts. DProd may also obtain a data encapsulation package that will be accompanying the key $pk_{\mathcal{C}}$ (this is a cryptographic engine that will enable the user to scramble the private information). At this stage DProd will be ready to submit the private data that will be encapsulated and submitted to DCol in enciphered form by the client's local host under the public-key $\mathsf{pk}_{\mathcal{C}}$.
CER processing stage. The CERs can be released by the primary service providers to the secondary market. A data market of CERs can be implemented at this stage that enables the exchange of CER records if this is desired (between secondary market entities). Data processors DProc will eventually form a database of CER records denoted by CDB. At this stage the CDB will be processed according to some prescribed *ciphertext-based* operations and the resulting scrambled outcome will be transmitted to the CAserver entities along with a request to release the appropriate information based on the contract. The contract authorities invoked by the action of DProc will inspect the submitted data for compliance to the contract and subsequently use its private key information to will facilitate the processing of the user data. The privacy safeguard of the system against illicit data usage is exactly at this stage where all data processing requests have to be authorized by the CAserver entities that check whether the contractual agreement is consistent with the intended processing operation that is requested by DProc. In the final step of the data-processing protocol, CAserver entities will return to DProc the value $f_{\mathcal{C}}(\boldsymbol{x})$ where $x = \langle x_1, \ldots, x_n \rangle \in (\mathsf{D}_{\mathcal{C}})^n$ are the data that were collected by n users under the contract $\mathcal{C}$.

In the basic framework as described above, the data processor DProc receives a cryptodatabase CDB that contains the encapsulated data that were submitted by the data-producers. While it is possible to submit the CDB directly to the CAserver entities for processing under the contract function $f_{\mathcal{C}}$, this poses two major shortcomings:

- The communication required to transfer CDB to CAserver's maybe disproportionately large compared to the required output of the data processing protocol.
- The computation cost imposed on CAserver's could be prohibitive as the entities in the worst case would have to decrypt all data and apply the $f_{\mathcal{C}}$ function to them in order to finish the protocol.

To put this into perspective consider the following setting: suppose that the data processor is interested in obtaining the mean salary of all the users in its cryptodatabase under a contract that allows the data-processor to do so provided that the salaries are revealed in batches of at least 1000 individual records. The data-processor may submit to the contract authority 1000 enciphered salary fields and the contract authority will decrypt the data, compute the mean and return it to the data processor. Clearly this solution is sub-optimal: (1) the communication is proportional to 1000 ciphertexts where the outcome of the protocol is a single element. (2) the computation that is imposed on `CAserver` entities is dependent to the size of all collected data where it would be preferable to be dependent only on the size of the output of the data processing stage.

To resolve the above problem we would like to devise methods that will allow to the data processor `DProc` to process the cryptodatabase *prior to submitting it* to the `CAserver` entities so that `DProc` can recover an object that will encapsulate the value of the mean (in this example) and this value can still be recovered by the contract authority following a protocol that has time and communication complexity proportional to the size of the average itself as opposed to proportional to the size of the cryptodatabase. This puts forth the following formalism:

Definition 1. *A contract function $f_{\mathcal{C}} : \mathbb{P}^* \rightarrow \mathbb{R}$ is said to support cryptodatabase processing under the encryption scheme $\langle \mathsf{gen}, \mathsf{enc}, \mathsf{dec} \rangle$ if the following condition holds: there exist two functions* `Combine` *and* `Reveal` *so that for any $n \in \mathbb{N}$ and any $m_1, \ldots, m_n \in \mathsf{D}_{\mathcal{C}}$, if $\langle pk, sk \rangle \leftarrow \mathsf{gen}(1^n)$, $c_i \leftarrow \mathsf{enc}(pk, m_i)$ for $i = 1, \ldots, n$ and $c \leftarrow \texttt{Combine}(c_1, \ldots, c_n)$, then we have $\texttt{Reveal}(sk, c) = f_{\mathcal{C}}(m_1, \ldots, m_n)$.*

Note that any function f supports cryptodatabase processing trivially under a public-key encryption scheme $\langle \mathsf{gen}, \mathsf{enc}, \mathsf{dec} \rangle$ by setting `Combine` to be the identity function and `Reveal` to simply decrypt each ciphertext individually and then apply the function $f_{\mathcal{C}}$ to the decrypted vector. Given this observation we will be interested in the following problem: given a function $f_{\mathcal{C}}$ find suitable encryption schemes under which the function $f_{\mathcal{C}}$ supports cryptodatabase processing so that (1) the size of the output of `Combine` is minimal (this will minimize the communication complexity between the data-collector and the data-producer), and (2) the time-complexity of `Reveal` is minimal (this will minimize the time complexity of the `CAserver` entities' side of the protocol).

The Data Market. The framework of PIP allows data-collectors to freely trade the encapsulated private data in a data-market. In this section we comment how it is possible to facilitate such data-market operation within our framework.

A challenge of treating (encapsulated) private data as a commodity is the fact that a data-collector is capable of faking data collection from users and accumulate a number of private data fields without actually collecting them from users. Subsequently through free trade he may exchange such corrupt data with actual data in the data-market.

One can tackle this problem as follows: prior to submitting the data the user will receive the direction from the contract specifications to submit the encapsulated data to a data-transfer transaction "watchdog" `DTWatch` that will validate the transaction. We stress that no data will be revealed to `DTWatch`; the communication to `DTWatch` will serve only as a leveraging measure to discourage fake data-creation by the data-producer. `DTWatch` will receive the encapsulated data and sign them. The data-collector will accept signed encapsulated data.

Each data-collector DCol records the encapsulated data and they become its property but now the private data cannot be modified, get corrupted or manufactured by the data-collector without going through DTWatch. Subsequently, DCol can enter the data into the data-market virtual network where the encapsulated data become a commodity that can be traded. A special entity called the data market watchdog DMWatch will check the validity of the signatures of the DTWatch and will only allow properly signed data to enter the market.

We point at this stage that the DTWatch and DMWatch may communicate in special occasions, and in addition to the above, DMWatch will verify any credentials attached to the encapsulated data as they were attached by the data-transfer-watchdog entity and it may request the revealing of relevant communication transcripts by DMWatch for comparison with the communication transcripts requested from DCol. The data-market-watchdog may also check the identity of DCol against a blacklisting database where previous offenders that have "poisoned" the data-market with illegal data will be entered. In this way DMWatch will protect the encapsulated data as a commodity by isolating misbehaving DCol's (we note nevertheless that always a determined DCol can subvert any data collection system by launching a "distributed data submission" attack if it has the necessary resources).

2.1 Homomorphic Encryption Schemes

We will employ two homomorphic encryption schemes that we describe briefly here. The first one is the Paillier encryption scheme [46]; the key generation process selects a large composite $n = pq$, where p, q are two prime numbers that are assumed to be hard to be recovered from n; additionally the Decisional Composite Residuosity assumption holds over $\mathbb{Z}^*_{n^2}$; in particular it is hard to distinguish between the uniform distribution over the whole group and the uniform distribution over the set of n-th residues in $\mathbb{Z}^*_{n^2}$. In this scheme, to encrypt a plaintext $m \in \mathbb{Z}_n$, the sender selects $r \in \mathbb{Z}^*_n$ and transmits the cipherext $(1+n)^m r^n \bmod n^2$. In order to recover a plaintext, the receiver first applies the Carmichael value λ that corresponds to n on the ciphertext; this results to the value $(1+n)^{\lambda m} \bmod n^2$; over the subgroup $\langle(1+n)\rangle$ in $\mathbb{Z}^*_{n^2}$ the discrete-logarithm problem is easy and as a result the value m is computable. The Paillier encryption is homomorphic with respect to addition over the plaintext group: indeed, given c_1, c_2 and two ciphertexts encrypting m_1, m_2 it is easy to verify that $c_1 \odot c_2 = c_1 \cdot c_2 = (1+n)^{m_1+m_2}(r^*)^n \bmod n^2$. The scheme can be turned into a threshold encryption scheme as shown in [16].

The second homomorphic encryption, we will employ is due to Boneh et al. [8]. The key-generation process chooses a bilinear group G that has order $N = pq$; the public-key of the system is set to $\langle G, N, g, g_p\rangle$, where g is a generator of G, and g_p is an element of order p over G. The secret-key is set to the factorization of n. To encrypt a message $m \in \{0, 1, \ldots, t\}$, the value $g^m g_p^r$ is computed where r is selected at random from $\mathbb{Z}_N$. To decrypt a message, the receiver raises the ciphertext c to p something that cancels the random component of the ciphertext and reveals the value g^{mp}. Subsequently, the value m can be computed by solving the discrete-logarithm over $\langle g \rangle$; given that it is not easy to compute discrete-logarithms over that group, the plaintext spaces would have to restricted to logarithmic length. The scheme is homomorphic with respect to addition in the same way that the Paillier encryption, introduced above, is homomorphic. Moreover, the scheme is homomorphic with respect to multiplication for a single operation. This is as follows: using the fact that there exists a bilinear map over G, we have that for two ciphertexts, c_1, c_2, that contain the plaintexts m_1, m_2, it is possible to compute $e(c_1, c_2) = e(g^{m_1} g_p^{r_1}, g^{m_2} g_p^{r_2}) = e(g, g)^{m_1 m_2 + q(r_1 m_2 + r_2 m_1) + q^2 r_1 r_2}$. It follows that processing $e(c_1, c_2)e(g_p, g_p)^s$ would result to a ciphertext encrypting $m_1 \cdot m_2$ (note that the multiplication is thought to be over the integers as long as the domain from which the plaintexts are drawn is selected to be suitably small). The cryptosystem can be ported to the threshold setting using the techniques of [21].

3 Constructing PIP systems for Statistical Data

In this section we will present two instantiations of the general framework of privacy preserving information processing motivated by statistics extraction from private numerical data.

3.1 PIP for computing the moments of a sample distribution

In this section we focus on numerical data and in particular how it is possible to extract statistics from the data that the DProd users contribute to draw conclusions about the population probability distribution; we will focus on a univariate analysis and in particular in extracting the central moments of the statistical distribution. Suppose that $x_1, \ldots, x_N$ are the data that are contributed by the data producers. The r-th central moment of the sample statistical distribution is defined as $\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^r$ where μ is the first moment that coincides with the sample mean. Computing central moments allows one to calculate k-statistics that are the unique symmetric unbiased estimators of the cumulants of the sample statistical distribution. Sample central moments can be computed easily based on power sums $P_r = \sum_{i=1}^{N} x_i^r$ so in this section we will focus on the computation of such power sums.

We will consider the following description for the contract $\mathcal{C}$ of this section:

> "The data requested entered in this field are numerical and describe the quantity $X \in \mathsf{D}$; they will be used for purpose Y and only statistical information of samples of N elements size will be revealed. The statistical information collected will allow the approximation of the statistical distribution up to degree $S \in \mathbb{N}$."

As an example, consider that X="salary", Range $= [10K \ldots, 500K]$, Y is "market research", and $N = 1000$. The degree S provided in the description of the contract will be the upper bound on the degree of the power sum that can be extracted from the collected data (i.e., only the power sums $P_1, \ldots, P_S$ will be computed by the data processor). This in turn bounds the information about the statistical distribution that is provided by the PIP system (e.g., if $S = 2$ only the mean and the standard deviation will be possible to be extracted). Based on the above we will be interested in the following problem: how is it possible to extract r-power sums for $r \in \{1, \ldots, S\}$ where $S \in \mathbb{N}$ is a parameter from the collected data. The system presented in this section will be based on homomorphic encryption and in particular on Paillier encryption [46] but a specialized encoding will be required. Below let $\langle \mathsf{gen}, \mathsf{enc}, \mathsf{dec} \rangle$ be the Paillier encryption. Note that the homomorphic encryption of the operation suggests that there exist an operation $\odot$ such that $\mathsf{enc}(m) \odot \mathsf{enc}(m') = \mathsf{enc}(m + m')$. The characteristics of our solution are as follows:

• During the data collection stage the data value $v \in \mathsf{D} \subseteq \mathbb{Z}$ will be encrypted as an integer using the $\mathsf{enc}(\cdot)$ Paillier encryption function as follows: $c = \langle c[1], \ldots, c[S] \rangle = \langle \mathsf{enc}(v), \mathsf{enc}(v^2), \ldots, \mathsf{enc}(v^S) \rangle$.

• During the data-processing stage `DProc` will use the following `Combine` function to process the cryptodatabase $c_1, \ldots, c_N$. Recall that each c_i is in fact a vector of the form $\langle c_i[1], \ldots, c_i[S] \rangle$. Using the homomorphic property of the encryption function, `DProc` calculates the power sum vector ciphertext

$$\langle \bar{c}_1, \ldots, \bar{c}_S \rangle = \langle \odot_{i=1}^{N} c_i[1], \ldots, \odot_{i=1}^{N} c_i[S] \rangle$$

Then, it submits the power vector ciphertext for processing and the `CAserver` entities will apply its secret-key to recover the power sums $\sum_{i=1}^{N} v_i^r$ for $r \in \{1, \ldots, S\}$. Note that this requires that the capacity of the Paillier encryption is at least $N \cdot (\max \mathsf{D})^S$ i.e., the capacity of the encryption summation register is of $\log N + S \cdot len$ bits where $len = \log_2 \max \mathsf{D}$ the size required to encode the maximum element of D as an integer.

Efficiency. We observe that the cryptodatabase processing that is performed using `Combine` reduces the size of the cryptodatabase from $N \cdot S$ ciphertexts to a vector of S ciphertexts, where each one is of length at least $\log N + S \cdot len$ bits. This makes the communication complexity of the data processing stage only polylog dependent to N (it is only $S \log N + S^2 \cdot len$) and thus very close to the optimal communication that is the output length of the `Reveal` function and equals $S \cdot len$ bits. Note that S is only a small constant parameter (e.g., it can be as small as $S = 2$ if one wishes to extract only the mean and the standard deviation of the sample).

Security. Since we use a specialized encoding for the numerical data (a vector of S ciphertexts each one containing consecutive powers of the data input) and it holds that the recovery of the power sums is dependent on conforming to this encoding, the computation relies on the fact that data producers `DProd` are following the encoding specifications. This can be ensured by requiring that `DProd` proves in zero-knowledge for a ciphertext vector $\langle c[1], \ldots, c[S] \rangle$ that the plaintext of $c[i]$ equals the plaintext of $c[i-1]$ times the plaintext of $c[1]$. This can be done efficiently by employing the proofs of knowledge of [23]. Ensuring that the data collector has submitted the homomorphic aggregation of N ciphertexts can be done by repeating the preprocessing computation at a later stage; for this purpose the aggregated ciphertext submitted to the `CAserver` entities and the actual crypto database can be stored for post-computation auditing purposes; misbehaving `DProc` will be caught by comparing the transmitted ciphertext to `CAserver` entities and the extracted ciphertexts from the data market. Note that data market data are assumed to be assigned in a way that a `DProc` cannot forge (unless it goes to substantial lengths in introducing a distributed set of data producers, cf. the data market discussion in section 2).

Based on the above we have the following (informally stated):

Theorem 1. *The* PIP *system presented above correctly computes power sums of the inputs of degree up to S and assuming the security of threshold Paillier encryption it preserves the privacy of the data providers.*

3.2 PIP of the Pearson correlation coefficient of two samples

In this section we show how it is possible to extend the PIP system of the previous section to bivariate correlational statistics focusing on the Pearson correlation coefficient which is used to estimate the the correlation of two random variables x, y. Recall that the correlation coefficient for a sample $\langle x_1, \ldots, x_N \rangle$ equals

$$r_{xy} = \frac{N \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{N \sum x_i^2 - (\sum x_i)^2} \sqrt{N \sum y_i^2 - (\sum y_i)^2}}$$

Evidently, it can be easily computed if one has the power sums P_1, P_2 for the variables x and y as well as the sum of products $\sum x_i y_i$. In order to achieve this type of contract-based computation in this section we employ another type of additive homomorphic encryption that enables the computation of one multiplication as well as unlimited additions that was proposed in [8]. In the encryption scheme of [8] (we refer to it also as BGN encryption) given $c = \mathsf{enc}(m)$ and $c' = \mathsf{enc}(m')$ one can compute a ciphertext of the form $c \odot c' = \mathsf{enc}(m + m)$ but also by changing the representation of ciphertexts to an equivalent one it is possible to compute a ciphertext of the form $\mathsf{enc}(m) \otimes \mathsf{enc}(m) = \overline{\mathsf{enc}}(m \cdot m')$. The disadvantage of the scheme in general is that requires $\mathcal{O}(\sqrt{\#\mathsf{D}})$ steps for decryption where $m \in \mathsf{D}$ using standard time-memory trade-off techniques; while this makes the decryption less efficient than that of Paillier's encryption that was employed in the previous section, the overhead is not substantial for our application domain. The capacity that we will require from the summation register of the encryption would be $\log N + 2 \cdot len$ where len is the size of the maximum element in the integer range the `DProd` select values.

The contract description that we will employ in this section will be of the following form:

> "The data requested entered in these two fields are numerical and describe the quantities $X \in \mathsf{D}_1, Y \in \mathsf{D}_2$; they will be used for purpose Z and only statistical information of samples of N elements size will be revealed as well as their correlation coefficient. The statistical information

collected will allow the approximation of the statistical distribution up to degree 2."

The construction will be built on top of that of section 3.1 with the following modifications:

• During the data collection stage the $x = \mathsf{data} \in \mathsf{D}_1$ and $y = \mathsf{data} \in \mathsf{D}_2$ will be encrypted as integers using the $\mathsf{enc}(\cdot)$ encryption function of [8] as follows : $c = \langle c[x,1], c[x,2], c[y,1], c[y,2]\rangle = \langle \mathsf{enc}(x), \mathsf{enc}(x^2), \mathsf{enc}(y), \mathsf{enc}(y^2)\rangle$.

• During the data-processing stage `DProc` will use the following `Combine` function to process the cryptodatabase $c_1, \ldots, c_N$. Each c_i is in fact a vector of the form $\langle c_i[x,1],\ c_i[x,2], c_i[y,1], c_i[y,2]\rangle$. Using the homomorphic property of the encryption function, `DProc` calculates the four power sum ciphertexts $\overline{c}[x,1] = \odot_{i=1}^{N} c_i[x,1]$, $\overline{c}[x,2] = \odot_{i=1}^{N} c_i[x,2]$, $\overline{c}[y,1] = \odot_{i=1}^{N} c_i[y,1]$, $\overline{c}[y,2] = \odot_{i=1}^{N} c_i[y,2]$. Finally using the multiplicative homomorphic property `DProc` computes $\overline{c}_{i,j} = c_i[x,1] \otimes c_j[y,1]$ and again by employing the additive homomorphic property `DProc` computes $\overline{c}_{x,y} = \odot_{i,j=1}^{N} \overline{c}_{i,j}$. `DProc` will submit to `CAserver` entities the five ciphertexts $\overline{c}[x,1], \overline{c}[x,2], \overline{c}[y,1], \overline{c}[y,2], \overline{c}_{x,y}$.

Using its secret-key `CAserver` entities will recover (1) the sums, (2) the power sums, (3) sum of products in time $\sqrt{\#\mathsf{D}_1} + \sqrt{\#\mathsf{D}_2}$, $\sqrt{\#\mathsf{D}_1} + \sqrt{\#\mathsf{D}_2}$ and $\sqrt{\#\mathsf{D}_1 \cdot \#\mathsf{D}_2}$ steps respectively.

Efficiency. The length of the communication required for the computation of the correlation coefficient is equal to four ciphertexts that each one is of size $\log N + 2 \cdot len$ which is asymptotically optimal given the output size of the `Reveal` function.

Security. As in the case of section 3.1 the `DProc` needs to perform a zero-knowledge proof to ensure that in the four ciphertexts $\langle c[x,1], c[x,2], c[y,1], c[y,2]\rangle$ that are submitted, the plaintext of $c[x,2]$ is the square of the plaintext of $c[x,1]$ and similarly for $c[y,2]$ and $c[y,1]$. This can be done efficiently using the techniques of [23].

Based on the above we have the following (informally stated):

Theorem 2. *The* PIP *system presented above correctly computes the Pearson correlation coefficient of the inputs and assuming the security of threshold BGN encryption it preserves the privacy of the data providers.*

References

1. Gagan Aggarwal, Nina Mishra, and Benny Pinkas. Secure computation of the k th-ranked element. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 40–55. Springer, 2004.
2. Rakesh Agrawal, Alexandre Evfimievski, and Ramakrishnan Srikant. Information sharing across private databases. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 86–97, New York, NY, USA, 2003. ACM Press.

3. Selim G. Akl and Peter D. Taylor. Cryptographic solution to a multilevel security problem. In *CRYPTO*, pages 237–249, 1982.
4. Michael Antecol and Becky Bermount. Wired teens aren't naive about online privacy, forrester research, july 24, 2001.
5. Donald Beaver. Commodity-based cryptography. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing (STOC '97)*, pages 446–455, New York, May 1997. Association for Computing Machinery.
6. Josh Cohen Benaloh and Moti Yung. Distributing the power of a government to enhance the privacy of voters (extended abstract). In *PODC*, pages 52–62, 1986.
7. Matt Blaze. A cryptographic file system for unix. In *CCS '93: Proceedings of the 1st ACM conference on Computer and communications security*, pages 9–16, New York, NY, USA, 1993. ACM Press.
8. Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC*, pages 325–341, 2005.
9. Justin Brickell and Vitaly Shmatikov. Efficient anonymity-preserving data collection. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 76–85. ACM, 2006.
10. Ran Canetti, Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Adaptive security for threshold cryptosystems. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 1999.
11. Ann Cavoukian and Tyler Hamilton. The privacy payoff, mcgraw-hill, 2002.
12. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24:84–88, February 1981.
13. Gerald C. Chick and Stafford E. Tavares. Flexible access control with master keys. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 316–322. Springer, 1989.
14. Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme (extended abstract). In *26th Annual Symposium on Foundations of Computer Science*, pages 372–382, Portland, Oregon, 21–23October 1985. IEEE.
15. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In Walter Fumy, editor, *Advances in cryptology — EUROCRYPT '97: International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11–15, 1997: proceedings*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118, pub-SV:adr, 1997. Springer-Verlag.
16. Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136. Springer, 2001.
17. Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.
18. Paul Feldman and Silvio Micali. Byzantine agreement in constant expected time (and trusting no one). In *26th Annual Symposium on Foundations of Computer Science (FOCS '85)*, pages 267–276, Los Angeles, Ca., USA, October 1985. IEEE Computer Society Press.

19. Paul Feldman and Silvio Micali. Optimal algorithms for byzantine agreement. In Richard Cole, editor, *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing*, pages 148–161, Chicago, IL, May 1988. ACM Press.
20. Yair Frankel, Peter Gemmell, Philip D. MacKenzie, and Moti Yung. Optimal resilience proactive public-key cryptosystems. In *FOCS*, pages 384–393, 1997.
21. Yair Frankel, Philip D. MacKenzie, and Moti Yung. Robust efficient distributed rsa-key generation. In *STOC*, pages 663–672, 1998.
22. Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2004.
23. Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*, pages 16–30, 1997.
24. Zvi Galil, Alain Mayer, and Moti Yung. Resolving message complexity of byzantine agreement and beyond. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science, FOCS'95 (Milwaukee, WI, October 23-25, 1995)*, pages 724–733, Los Alamitos-Washington-Brussels-Tokyo, 1995. IEEE Computer Society, IEEE Computer Society Press.
25. Juan Garay and Yoram Moses. Fully polynomial byzantine agreement in $t+1$ rounds. In Alok Aggarwal, editor, *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pages 31–41, San Diego, CA, USA, May 1993. ACM Press.
26. Eu-Jin Goh, Hovav Shacham, Nagendra Modadugu, and Dan Boneh. Sirius: Securing remote untrusted storage. In *NDSS*. The Internet Society, 2003.
27. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229. ACM, 1987.
28. Shafi Goldwasser. Multi-party computations: Past and present. In *PODC*, pages 1–6, 1997.
29. Philippe Golle, Frank McSherry, and Ilya Mironov. Data collection with self-enforcing privacy. *ACM Trans. Inf. Syst. Secur.*, 12(2), 2008.
30. Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT ' 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 539–556, Brugge, Belgium, 2000. Springer-Verlag, Berlin Germany.
31. Statistical Research Inc. How people use the internet 2001, study, june 2001.
32. Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca N. Wright. A new privacy-preserving distributed k-clustering algorithm. In Joydeep Ghosh, Diane Lambert, David B. Skillicorn, and Jaideep Srivastava, editors, *SDM*. SIAM, 2006.
33. M. Jakobsson. A practical mix. In *Advances in Cryptology – EUROCRYPT '98*, pages 448–461, 1998.
34. Murat Kantarcioglu and Chris Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Trans. Knowl. Data Eng.*, 16:1026–1037, 2004.
35. Jonathan Katz, Steven Myers, and Rafail Ostrovsky. Cryptographic counters and applications to electronic voting. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 78–92. Springer, 2001.

36. Aggelos Kiayias and Antonina Mitrofanova. Testing disjointness of private datasets. In *Financial Cryptography*, pages 109–124, 2005.
37. Aggelos Kiayias and Antonina Mitrofanova. Syntax-driven private evaluation of quantified membership queries. In *4th International Conference on Applied Cryptography and Network Security (ACNS'06)*, 2006.
38. Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. In *Advances in Cryptology - EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques*, pages 571 – 589, 2004.
39. Aggelos Kiayias and Moti Yung. Self-tallying elections and perfect ballot secrecy. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*, volume 2274 of *Lecture Notes in Computer Science*, pages 141–158. Springer, 2002.
40. Aggelos Kiayias and Moti Yung. Non-interactive zero-sharing with applications to private distributed decision making. In Rebecca N. Wright, editor, *Financial Cryptography, 7th International Conference, FC 2003, Guadeloupe, French West Indies, January 27-30, 2003, Revised Papers*, volume 2742 of *Lecture Notes in Computer Science*, pages 303–320. Springer, 2003.
41. Aggelos Kiayias and Moti Yung. The vector-ballot e-voting approach. In Ari Juels, editor, *Financial Cryptography, 8th International Conference, FC 2004, Key West, FL, USA, February 9-12, 2004. Revised Papers*, volume 3110 of *Lecture Notes in Computer Science*, pages 72–89. Springer, 2004.
42. Lea Kissner and Dawn Xiaodong Song. Privacy-preserving set operations. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2005.
43. Yehuda Lindell and Benny Pinkas. Privacy preserving data mining. In *CRYPTO '00: Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, pages 36–54, London, UK, 2000. Springer-Verlag.
44. Paul F. Nunes and Ajit Kambil. Internet privacy: A look under the covers, accenture institute for strategic change, july 2000; www.accenture.com.
45. Christine M. O'Keefe, Ming Yung, Lifang Gu, and Rohan Baxter. Privacy-preserving data linkage protocols. In *WPES '04: Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 94–102, New York, NY, USA, 2004. ACM Press.
46. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. *Lecture Notes in Computer Science*, 1592:223–238, 1999.
47. Indrakshi Ray, Indrajit Ray, and Natu Narasimhamurthi. A cryptographic solution to implement access control in a hierarchy and more. In *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*, pages 65–73, New York, NY, USA, 2002. ACM Press.
48. Kazue Sako and Joe Kilian. Secure voting using partially compatible homomorphisms. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 411–424. Springer, 1994.
49. Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme: A practical solution to the implementation of a voting booth. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology—EUROCRYPT 95*, volume

921 of *Lecture Notes in Computer Science*, pages 393–403. Springer-Verlag, 21–25 May 1995.

50. Ravi S. Sandhu. Cryptographic implementation of a tree hierarchy for access control. *Inf. Process. Lett.*, 27(2):95–98, 1988.
51. A. Shamir. How to share a secret. *Communications of the ACM*, 22(11), 1979.
52. Nick Szabo. The idea of smart contracts. http://szabo.best.vwh.net/smart_contracts_idea.html, 1997.
53. Jaideep Vaidya and Chris 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*, pages 639–644, New York, NY, USA, 2002. ACM Press.
54. Zhiqiang Yang and Rebecca N. Wright. Privacy-preserving computation of bayesian networks on vertically partitioned data. *IEEE Trans. Knowl. Data Eng.*, 18(9):1253–1264, 2006.
55. Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. Anonymity-preserving data collection. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 334–343, New York, NY, USA, 2005. ACM Press.
56. Sheng Zhong, Zhiqiang Yang, and Rebecca N. Wright. Privacy-enhancing - anonymization of customer data. In Chen Li, editor, *PODS*, pages 139–147. ACM, 2005.