Published in A.-R. Sadeghi, Ed., *Financial Cryptography and Data Security (FC 2013)*, vol. 7879 of *Lecture Notes in Computer Science*, pp. 111–125, Springer, 2013.

A Scalable Scheme for Privacy-Preserving Aggregation of Time-Series Data

Marc Joye and Benoît Libert

Technicolor
975 avenue des Champs Blancs, 35576 Cesson-Sévigné Cedex, France
{marc.joye,benoit.libert}@technicolor.com

Abstract. Suppose that a set of multiple users uploads in every time period encrypted values of some data. The considered problem is how an *untrusted* data aggregator can compute the sum of all users' values but nothing more. A solution was recently given by Shi *et al.* (NDSS 2011). However, as advocated by the authors, the proposed encryption scheme suffers from some limitations. In particular, its usage is restricted to small plaintext spaces. This paper presents a practical scheme which, advantageously, can accommodate large plaintext spaces. Somewhat surprisingly, it comes with an efficient security reduction, regardless of the number of users. Furthermore, the proposed scheme requires a minimal number of interactions, is efficient for both encryption and decryption/aggregation and can operate in an off-line/on-line mode.

Keywords: Private aggregation, smart metering, homomorphic encryption, large data sets.

1 Introduction

A fundamental problem is that of private data analysis where a third party has to compute some aggregate statistics over some sensitive data held by individuals. This problem finds concrete applications in a number of situations. When the third party, called hereafter *aggregator*, is trusted an easy solution would be to ask the users to encrypt their data using the aggregator's public key. Upon receiving the ciphertexts the aggregator applies its private key to recover the data in clear and then compute statistics. The problem becomes much more challenging in the case of an *untrusted* aggregator. This is the setting we are dealing with in this paper.

As an illustrative example, consider the case of smart energy metering (e.g., gas, electricity or water). Frequent aggregates of consumption over a population of users is very useful to finely tune the service by adapting the load or forecasting the supply, which results in better prices. It also reveals useful to rapidly detect anomalies on the grid in the case of accidental leakage. While the computation of aggregate statistics is beneficial to the consumers, it may legitimately raise privacy concerns. Electricity smart meters typically report the electricity usage every 15 minutes. Such data can be used to infer information

about users' behaviors (e.g., when they are at home and what they do). Other examples include collecting medical data for disease monitoring or developing new drugs, collecting users' preferences for recommendation systems, etc.

The previous examples clearly highlight the need of allowing users to privately share their data while at the same time allowing a [non-necessarily trusted] aggregator to carry out aggregate statistics. Different types of statistics can be computed over private data. Sums and averages are widespread examples.

Privacy-preserving protocols. Several approaches can be found in the literature (see [18] for a comprehensive survey of these) to address the problem of computing the sum of time-series data, in a privacy-preserving fashion.[1]

Borrowing the taxonomy of [18], general privacy-preserving protocols can be characterized from the following dimensions:

- *Aggregate function:* This is the function evaluated by the aggregator. In our case, the function is the sum of the users' private inputs.
- *User synchronization:* A protocol is said asynchronous if the users can report their data independently of each other to the aggregator and at any time. Following the terminology of [18], our protocol is synchronous as we consider time-series data aggregation and so users should report their data at approximatively the same time (*i.e.*, the time when aggregation is computed). However, it does *not* require interaction among users. In that sense, the protocol we propose in this paper is synchronous yet non-interactive.
- *Fault tolerance:* This notion captures the capability of coping with failures. Our basic protocol assumes that there are no failures: all users report their data. This basic protocol can nevertheless be adapted to support failures; see [8, 17].
- *Communication model:* The communication between the users and the aggregator can be unidirectional or bidirectional. The bidirectional setting requires a return channel from the aggregator to the users or among users. Our protocol merely requires an unidirectional communication link to the aggregator.
- *Privacy notions:* There are different flavors of privacy. Our protocol provably achieves the strongest notion of aggregator-obliviousness: The aggregator learns nothing beyond the output of the aggregate function. See Section 3.
- *Aggregate error:* The output of the aggregate function can be exact or noisy. Our basic protocol returns the exact sum of all users' data. It can however be modified to return a noisy sum through differential-privacy techniques, similarly to [25].
- *Group management:* This last notion indicates if the protocol is static or dynamic. Dynamic protocols allow users to join or leave the group without requiring a new set-up phase. As described, our protocol is static. Techniques for dealing with dynamic settings can be found in [8, 17].

[1] We note that very efficient solutions are known, based on symmetric-key cryptography, in the case of a *trusted* aggregator. See e.g. [7].

Related work. In [24], Rastogi and Nath suggested that each user encrypts her private data using an additively homomorphic encryption scheme. The aggregator collects all the ciphertexts, aggregates them to get the encryption of the aggregate sum and sends this value back to the users. All the users, who possess each a share of the decryption key, contribute to the decryption of the aggregate sum by sending their decryption share to the aggregator. The aggregator then combines all decryption shares received from the users to get in clear the sum of the users' private data.

Like other proposals [1, 20, 21], the approach of [24] requires a bidirectional communication link between the users and the aggregator. Unfortunately a return channel is not always available. To alleviate this issue, Garcia and Jacobs [15] came up with an aggregation protocol based on homomorphic encryption and additive secret sharing. Their proposal eliminates the need for a bidirectional channel. On the downside, the communication complexity is quadratic in the number of users and each user has to compute a linear number of homomorphic encryptions.

Independently, Kursawe *et al.* [19] described four protocols allowing the aggregation of users' data and comparisons among data aggregates without using bidirectional channels. Nevertheless, the first three protocols incur interaction, Diffie-Hellman key agreements or bilinear map evaluations. They thus require either user-to-user communication or somewhat costly arithmetic operations. The fourth protocol of [19] has low overhead but, like the previous ones, it requires each user to store the fixed public key of all other users.

Back in 2011, Shi, Chan, Rieffel, Chow and Song [25] described a completely non-interactive solution. They astutely suggested to split the aggregator capability, viewed as an un-blinding key s_0, into additive shares among the set of users, say $s_0 = \sum_i s_i$. Each user makes use of her secret share s_i and blinds her private data $x_{i,t}$ at time period t with a one-time mask derived from s_i and t to obtain a masked value $c_{i,t}$. At each time period, the aggregator collects all the $c_{i,t}$'s. The difficulty resides in finding a scheme such that the aggregation results in the private data adding together and the masks summing up to a value depending only on t and on $\sum_i s_i = s_0$. Using the un-blinding key s_0, the aggregator can so recover in clear the sum of the users' private data. Shi *et al.* [25] gave a solution that generically works in any prime order group where the Decision Diffie-Hellman (DDH) assumption holds. They also provided a security proof in a formally defined privacy model. See § 3.2 for a detailed description of their scheme. The authors however left open a couple of research challenges (cf. [25, Section 8]). One of them is that of user failure (fault tolerance) and efficient support of dynamic joins and leaves. Solutions to this problem have been found in [1, 8, 17].

Among the fault-tolerant systems [1, 8, 17], the construction of Ács and Castellucia [1] requires shared keys among all pairs of users, which incurs a linear storage in the size of the network at each user. Moreover, users should also be able to receive messages from the aggregator. In a recent work, Jawurek and Kerschbaum [17] managed to avoid these overheads in a protocol that further

computes *weighted* sums. On one hand, their protocol allows for an arbitrary number of missing inputs from users. It also eliminates the need for synchronization among these. On the other hand, it involves distributed key managing authorities whose task is to jointly decrypt Paillier encryptions during the protocol. As a result, these key managing authorities have to be involved—and interact with the user performing the calculation—in each aggregation operation.

Our contribution. In this paper, we reconsider the fully non-interactive construction of Shi *et al.* [25] and show how to get rid of one of its limitations. An important challenge left open in [25] (and also discussed in [8]) is that of efficiently computing sums over large plaintext spaces. This paper proposes a scheme that supports large plaintext spaces and number of users. At the same time, the decryption algorithm operates in constant time, regardless of the number of users. As a bonus, the scheme also provides a great on-line/off-line efficiency: namely, using pre-computations, the encryptor is left with a mere modular multiplication in the on-line phase (*i.e.*, when the data to be encrypted is known), which is highly desirable when computations take place on resource-limited devices.

To achieve this, we follow the research direction suggested in [25], namely investigating other algebraic settings. The natural candidate is the Paillier cryptosystem [23]. However, the proof given in [25] does not seem to readily extend to this setting due to technical difficulties arising when we want to rely on the DDH assumption. The main hurdle appears to find a way to efficiently hash to a cyclic subgroup of hidden order while remaining able—in order to properly simulate the adversary's view in the security proof—to explain hash values as being obtained by exponentiating the output of a random oracle whose range has no specific algebraic structure. While several techniques are known (e.g., [5, 16]) to efficiently hash onto cyclic groups of public order, this turns out to be more complicated in DDH-hard groups of hidden order like the subgroup of squares modulo N or N^2, for some moduli $N = pq$ of unknown factorization. By trading the DDH assumption for the Decision Composite Residuosity (DCR) assumption [23], we eliminate the need to hash onto a group of hidden order. Somewhat surprisingly, we are moreover able to derive a security proof with a much tighter (*i.e.*, independent of the number of users) reduction in the random oracle model [2]. This in turn results in better parameters when the scheme is concretely instantiated.

While practical, our scheme inherits the limitations of [25] in that it does not allow partial aggregations in the presence of missing contributions. Unlike [20, 21, 17], it does not extend to compute weighted sums and, unlike [21, 17], it requires synchronization among participants. On the other hand, our system inherits the main advantage of [25]: the data aggregator can process users' data by itself without having to interact with a distributed key-managing authority at each aggregation. Here, the trusted party that generates the Paillier public key is never involved in any aggregation and can remain off-line after the setup phase. As for its advantages over [1], our construction only requires users to store $O(1)$ values and does not require a bidirectional channel.

Like [24, 25], the proposed scheme can serve as a building block for the fault-tolerant solution of [8] while enjoying the benefits of our construction. In fact, all the extensions of [25] are also possible with our system. In particular, although the focus of this paper is put on the encryption, the proposed scheme is compatible with the differential-privacy framework [13, 12]. In this case, the users simply need to add some appropriately-generated noise to their data prior to encryption.

Outline of the paper. The rest of this paper is organized as follows. The next section introduces some mathematical background and related cryptographic problems. Section 3 explains what is meant by aggregator obliviousness. Section 4 is the core of the paper. It presents a new aggregator-oblivious encryption scheme, a security proof of which is provided in Section 5. Finally, Section 6 discusses the performance of the proposed scheme and some implementation issues.

2 Preliminaries

In this section we review the necessary background and introduce some notation used throughout the paper.

2.1 N-th residues and discrete logarithms in $(\mathbb{Z}/N^2\mathbb{Z})^*$

Let p be a prime. Consider the ring $\mathbb{Z}/p^2\mathbb{Z} = \{0, 1, \ldots, p^2 - 1\}$. Its multiplicative group is given by $(\mathbb{Z}/p^2\mathbb{Z})^* = \{x \in \mathbb{Z}/p^2\mathbb{Z} \mid \gcd(x, p) = 1\}$. This group has $p(p-1)$ elements. Letting $\mathrm{ord}(x)$ denote the order of x as an element of $(\mathbb{Z}/p^2\mathbb{Z})^*$ (*i.e.*, the smallest nonnegative integer α such that $x^\alpha \equiv 1 \pmod{p^2}$), Lagrange's theorem tells that $\mathrm{ord}(x)$ divides $p(p-1)$. In particular, $(\mathbb{Z}/p^2\mathbb{Z})^*$ has a p-order subgroup Γ_p given by

$$\Gamma_p = \{x \in (\mathbb{Z}/p^2\mathbb{Z})^* \mid x^p \equiv 1 \pmod{p^2}\} \ .$$

Suppose now that $x = 1 + \beta p$ with $\beta \in \{0, \ldots, p-1\}$ —observe that such an x is in $(\mathbb{Z}/p^2\mathbb{Z})^*$ since $\gcd(1 + \beta p, p) = 1$. Further, the binomial identity yields

$$(1 + \beta p)^t \equiv \sum_{k=0}^{t} \binom{t}{k} 1^{n-k} (\beta t)^k \equiv \binom{t}{0} + \binom{t}{1} \beta p \equiv 1 + t\beta p \pmod{p^2} \ .$$

Hence, we have $(1 + \beta p)^p \equiv 1 \pmod{p^2}$ and consequently $1 + \beta p \in \Gamma_p$. From $\Gamma_p \cong (\mathbb{Z}/p\mathbb{Z})^+$ (a cyclic group), we also deduce that any element $x \neq 1 \in \Gamma_p$ generates it. In particular, $1 + p$ is a generator of Γ_p. As a result, we get

$$\Gamma_p = \{(1+p)^\beta \pmod{p^2} \mid \beta \in \{0, \ldots, p-1\}\} = \{1 + \beta p \mid \beta \in \{0, \ldots, p-1\}\} \ .$$

Let now $N = pq$ where p and q are (distinct) primes. By Chinese remaindering, we can construct the multiplicative group $(\mathbb{Z}/N^2\mathbb{Z})^* \cong (\mathbb{Z}/p^2\mathbb{Z})^* \times (\mathbb{Z}/q^2\mathbb{Z})^*$. This group has order $\#(\mathbb{Z}/N^2\mathbb{Z})^* = \#(\mathbb{Z}/p^2\mathbb{Z})^* \times \#(\mathbb{Z}/q^2\mathbb{Z})^* = p(p-1)q(q-1) =$

$N\phi(N)$ where $\phi(N) = (p-1)(q-1)$ is Euler's totient function. Similarly, we can define $\Gamma_N \cong \Gamma_p \times \Gamma_q$ with $\#\Gamma_N = \#\Gamma_p \times \#\Gamma_q = pq = N$ as

$$\begin{aligned}\Gamma_N &= \{(1+N)^\beta \pmod{N^2} \mid \beta \in \{0,\ldots,N-1\}\} \\ &= \{1+\beta N \mid \beta \in \{0,\ldots,N-1\}\} \ .\end{aligned}$$

Again, it is worth noticing that $(1+N)^\beta \bmod N^2 = 1+\beta N$; note also that β is defined modulo N. This shows that computing discrete logarithms in Γ_N is easy. Namely, given two random elements $u, v \in \Gamma_N$, it is easy to find α such that $v = u^\alpha \pmod{N^2}$:

$$\alpha = \frac{L(v)}{L(u)} \bmod N \quad \text{where } L(x) = \frac{x-1}{N} \ .$$

Proof. Since $u \in \Gamma_N$, we have $u \equiv 1 \pmod{N}$ and so we can write $u = 1+\beta N$ where $\beta = (u-1)/N = L(u)$. Likewise, we can write $v = 1 + L(v)N$. Since we have that $u^\alpha = v$, it follows that $(1+L(u)N)^\alpha \equiv 1 + L(v)N \pmod{N^2} \iff 1+\alpha L(u)N \equiv 1+L(v)N \pmod{N^2} \iff N(\alpha L(u) - L(v)) \equiv 0 \pmod{N^2}$. Since $N(\alpha L(u) - L(v)) \bmod N^2 = N[(\alpha L(u) - L(v)) \bmod N]$, we get $\alpha L(u) \equiv L(v) \pmod{N}$ and consequently $\alpha = L(v)/L(u) \bmod N$. □

Another subgroup of $(\mathbb{Z}/N^2\mathbb{Z})^*$ is the *group of N-th residues* given by

$$\mathfrak{S}_N = \{x^N \pmod{N^2} \mid x \in (\mathbb{Z}/N^2\mathbb{Z})^*\} \ .$$

This subgroup has order $\phi(N)$ and computing discrete logarithms in $\mathfrak{S}_N$ is as hard as computing discrete logarithms in $(\mathbb{Z}/N\mathbb{Z})^*$ (see [22, 26]). Given two elements $u, v \in \mathfrak{S}_N$ with $v = u^\alpha \pmod{N^2}$ for some $\alpha \in \{0,\ldots,\phi(N)\}$, it turns out that $v \equiv u^\alpha \pmod{N}$. It is also worth observing that each N-th residue possesses exactly N roots of order N, of which only one is smaller than N. Indeed, if $y \equiv x^N \pmod{N}$ then so is $y \equiv (x_0 + \beta N)^N \pmod{N^2}$ with $x_0 := x \bmod N$ and $\beta \in \{0,\ldots,N-1\}$.

2.2 DCR complexity assumption

We need the following hardness assumption, introduced in [23], which is implied by the commonly assumed intractability of factoring.

Definition 1. *Let $N = pq$ be a product of two large primes. The* Decision Composite Residuosity (DCR) *assumption in $(\mathbb{Z}/N^2\mathbb{Z})^*$ is to distinguish among the following two distributions given only $N = pq$:*

$$D_0 = \{x^N \pmod{N^2} \mid x \in_R (\mathbb{Z}/N^2\mathbb{Z})^*\} \quad \textit{and} \quad D_1 = \{x \in_R (\mathbb{Z}/N^2\mathbb{Z})^*\} \ .$$

The DCR assumption *states that the advantage of any distinguisher $\mathcal{D}$, defined as the distance*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{DCR}}(\mathcal{D}) = \big|&\Pr[\mathcal{D}(y,N) = 1 \mid y = x^N \bmod N^2, x \in_R (\mathbb{Z}/N\mathbb{Z})^*] \\ &- \Pr[\mathcal{D}(y,N) = 1 \mid y \in_R (\mathbb{Z}/N^2\mathbb{Z})^*]\big|\end{aligned}$$

where probabilities are taken over all coin tosses, is a negligible function.

3 Aggregator-Oblivious Encryption

We start with the definition and then proceed with the corresponding security notion. We refer the reader to [25] for further introductory background.

Definition 2. *An* aggregator-oblivious encryption scheme *is a tuple of algorithms,* $(\mathsf{Setup}, \mathsf{Enc}, \mathsf{AggrDec})$*, defined as:*

$\mathsf{Setup}(1^\kappa)$ *On input a security parameter* κ*, a trusted dealer generates the system parameters* param*, the aggregator's private key* sk_0*, and the private key* sk_i *for each user* i $(1 \leq i \leq n)$*;*

$\mathsf{Enc}(\mathsf{param}, \mathsf{sk}_i, x_{i,t})$ *At time period* t*, user* i *encrypts a value* $x_{i,t}$ *using her private encryption key* sk_i *to get* $c_{i,t} = \mathsf{Enc}(\mathsf{param}, \mathsf{sk}_i, x_{i,t})$*.*

$\mathsf{AggrDec}(\mathsf{param}, \mathsf{sk}_0, c_{1,t}, \ldots, c_{n,t})$ *At time period* t*, the aggregator using* sk_0 *obtains* $X_t = \sum_{i=1}^{n} x_{i,t}$ *as* $X_t = \mathsf{AggrDec}(\mathsf{param}, \mathsf{sk}_0, c_{1,t}, \ldots, c_{n,t})$*.*

3.1 Aggregator obliviousness

Basically, the security notion of *aggregator obliviousness* (AO) requires that the aggregator cannot learn, for each time period, anything more than the aggregate value X_t from the encrypted values of n (honest) users. If there are corrupted users (*i.e.*, users sharing their private information with the aggregator), the notion only requires that the aggregator gets no extra information about the values of the honest users beyond that their aggregate value. Furthermore, it is assumed that each user encrypts only one value per time period.

More formally, AO is defined by the following game between a challenger and an attacker.

Setup The challenger runs the Setup algorithm and gives param to the attacker.

Queries In a first phase, the attacker can submit queries that are answered by the challenger. The attacker can make two types of queries:

1. Encryption queries: The attacker submits $(i, t, x_{i,t})$ for a pair (i, t) and gets back the encryption of $x_{i,t}$ under key sk_i for time period t;
2. Compromise queries: The attacker submits i and receives the private key sk_i of user i; if $i = 0$, the attacker receives the private key of the aggregator.

Challenge In a second phase, the attacker chooses a time period $t^\star$. Let $\mathcal{U}^\star \subseteq \{1, \ldots, n\}$ be the whole set of users for which, at the end of the game, no encryption queries have been made on time period $t^\star$ and no compromise queries have been made. The attacker chooses of a subset $\mathcal{S}^\star \subseteq \mathcal{U}^\star$ and two different series of triples

$$\langle (i, t^\star, x_{i,t^\star}^{(0)}) \rangle_{i \in \mathcal{S}^\star} \quad \text{and} \quad \langle (i, t^\star, x_{i,t^\star}^{(1)}) \rangle_{i \in \mathcal{S}^\star} ,$$

that are given to the challenger. Further, if the aggregator capability sk_0 is compromised at the end of the game and $\mathcal{S}^\star = \mathcal{U}^\star$, it is required that

$$\sum_{i \in \mathcal{S}^\star} x_{i,t^\star}^{(0)} = \sum_{i \in \mathcal{S}^\star} x_{i,t^\star}^{(1)} \ . \tag{1}$$

Guess The challenger chooses at random a bit $b \in \{0,1\}$ and returns the encryption of $\langle x_{i,t^\star}^{(b)} \rangle_{i \in \mathcal{S}^\star}$ to the attacker.

More queries The attacker can make more encryption and compromise queries.

Outcome At the end of the game, the attacker outputs a bit b' and wins the game if and only if $b' = b$. As usual, $\mathcal{A}$'s advantage is defined to be

$$\mathbf{Adv}^{\mathsf{AO}}(\mathcal{A}) := |\Pr[b' = b] - 1/2|.$$

Remark 1. Note in the "More queries" stage that since $\mathcal{S}^\star \subseteq \mathcal{U}^\star$, the attacker cannot submit an encryption query $(i, t^\star, \cdot)$ with $i \in \mathcal{S}^\star$ or a compromise query i with $i \in \mathcal{S}^\star$.

Definition 3. *An encryption scheme is said to meet the* AO security notion *if no probabilistic polynomial-time attacker can guess correctly in the above game the bit b with a probability non-negligibly better (in the security parameter) than* $1/2$. *The probability is taken over the random coins of the game according to the distribution induced by* Setup *and over the random coins of the attacker.*

3.2 Shi *et al.*'s scheme

In [25], Shi, Chan, Rieffel, Chow and Song consider the following encryption scheme. They show that the scheme meets the AO security notion under the DDH assumption [4], in the random oracle model.

$\mathsf{Setup}(1^\kappa)$ Let a group $\mathbb{G}$ of prime order q for which the DDH assumption holds, and let a random generator $g \in \mathbb{G}$. Let also a hash function $H : \mathbb{Z} \to \mathbb{G}$ viewed as a random oracle. Finally, let n random elements in $\mathbb{Z}/q\mathbb{Z}$, $s_1, \ldots, s_n$, and define $s_0 = -\sum_{i=1}^n s_i \bmod q$.

$\mathsf{param} = \{\mathbb{G}, g, H\}$; $\mathsf{sk}_i = s_i$ (for $0 \leq i \leq n$).

$\mathsf{Enc}(\mathsf{param}, \mathsf{sk}_i, x_{i,t})$ At time period t, for a private input $x_{i,t} \in \mathbb{Z}/q\mathbb{Z}$, user i produces

$$c_{i,t} = g^{x_{i,t}} H(t)^{s_i} \quad .$$

$\mathsf{AggrDec}(\mathsf{param}, \mathsf{sk}_0, c_{1,t}, \ldots, c_{n,t})$ The aggregator obtains the sum X_t for time period t by first computing $V_t := H(t)^{s_0} \prod_{i=1}^n c_{i,t} = g^{X_t}$ and next the discrete logarithm of V_t w.r.t. basis g.

Remark 2. Since g has order q, note that the so-obtained value for X_t is defined modulo q.

We see that in the AggrDec algorithm the aggregator has to compute the value of X_t from $V_t = g^{X_t}$ in $\mathbb{G}$. For known groups satisfying Shi *et al.*'s setting (*i.e.*, prime-order DDH groups), the most appropriate method is Pollard's λ algorithm (or variants thereof) and requires that the range of X_t is small. In the next section, we present a scheme where the computation of discrete logarithms can be done efficiently without such a restriction on the range, while at the same time, meeting the AO security notion.

4 New Scheme

Shi *et al.*'s scheme involves the computation of a discrete logarithm in a prime-order group for which the DDH assumption holds. We rather consider groups $\mathbb{G}$ of composite order for which there is a subgroup $\mathbb{G}_1$ [of unknown order] wherein some intractability assumption holds and another subgroup $\mathbb{G}_2$ wherein discrete logarithms are easily computable. It is crucial that the order of $\mathbb{G}_1$, $\#\mathbb{G}_1$, is only known to a trusted dealer. For anyone else (including the aggregator), $\#\mathbb{G}_1$ must remain unknown. Merely an upper bound on $\#\mathbb{G}_1$ can be derived. We present below a scheme fulfilling these requirements.

$\mathsf{Setup}(1^\kappa)$ On some input security parameter κ, the trusted dealer randomly generates a modulus $N = pq$, which is the product of two equal-size primes p, q. Note that size condition on p and q implies that $\gcd(\phi(N), N) = 1$. It also defines a hash function $H : \mathbb{Z} \to (\mathbb{Z}/N^2\mathbb{Z})^*$ that will be viewed as a random oracle in the security analysis. Letting ℓ the bit-length of N, from n randomly chosen elements in $\pm\{0,1\}^{2\ell}$, $s_1, \ldots, s_n$, it finally sets $s_0 = -\sum_{i=1}^n s_i$ and defines $\mathsf{param} = \{N, H\}$ as well as $\mathsf{sk}_i = s_i$ (for $0 \leq i \leq n$).

$\mathsf{Enc}(\mathsf{param}, \mathsf{sk}_i, x_{i,t})$ At time period t, for a private input $x_{i,t} \in \mathbb{Z}/N\mathbb{Z}$, user i produces

$$c_{i,t} = (1 + x_{i,t}N) \cdot H(t)^{s_i} \bmod N^2 \ .$$

$\mathsf{AggrDec}(\mathsf{param}, \mathsf{sk}_0, c_{1,t}, \ldots, c_{n,t})$ The aggregator obtains the sum X_t for time period t by first computing $V_t := H(t)^{s_0} \prod_{i=1}^n c_{i,t} \bmod N^2$ and next X_t as

$$X_t = \frac{V_t - 1}{N} \ .$$

The correctness follows by observing that

$$H(t)^{s_0} \prod_{i=1}^n c_{i,t} \equiv \prod_{i=1}^n (1 + x_{i,t}N) \equiv 1 + \big(\sum_{i=1}^n x_{i,t} \bmod N\big)N \pmod{N^2} \ .$$

Again, observe that the value of X_t is defined modulo N. Hence, if $\sum_{i=1}^n x_{i,t} < N$, we have $X_t = \frac{V_t - 1}{N} = \sum_{i=1}^n x_{i,t}$ over the integers. The main difference with the scheme of Shi *et al.* resides in that there is no discrete logarithm to compute in a group in which the DDH assumption holds. On the contrary, the recovery of X_t from the accumulated product, V_t, is now easy. As a result, there is no longer the restriction on the size of $x_{i,t}$ or on the total number n of users, as long as $\sum_{i=1}^n x_{i,t} < N$. Typically, N is a 2048-bit value. In practice, there is therefore no restriction.

Another interesting advantage of the scheme over [25] is that it allows efficiently encrypting "on-the-fly" [14]. Namely, exponentiations $H(t)^{s_i} \bmod N^2$ can be pre-computed in such a way that, when the plaintext $x_{i,t}$ is known, the sender only has to compute a modular multiplication to get $c_{i,t}$. In applications to smart metering systems, this advantage may be crucial as computations usually take place in constrained devices.

One surprising thing about the scheme is its extreme simplicity. Indeed, somewhat unexpectedly, we do not even need to restrict the range of the random oracle H to, for example, the subgroup of quadratic residues in $(\mathbb{Z}/N^2\mathbb{Z})^*$ or the subgroup of elements with positive Jacobi symbol modulo N. If $H(t) \bmod N$ has Jacobi symbol -1 modulo N, the parity of s_i is leaked by $c_{i,t}$. However, as will be established in Section 5, this does not jeopardize the security of the scheme as long as the values $s_i \bmod N$ remain computationally hidden.

5 Security Proof

Although the scheme is very similar to the construction of Shi *et al.*, its security analysis is completely different (and actually simpler). Albeit taking place in the random oracle model [2], it bears similarities with the techniques of Cramer and Shoup [10] (see also [6]) in that it appeals to an information theoretic argument exploiting some entropy hidden in the private key.

As a result, we obtain a much tighter security reduction than in [25]. Here, the gap between the DCR distinguisher and the adversary's advantage is only proportional to the number of encryption queries. In contrast, the construction in [25] suffers from an additional degradation factor of $O(n^3)$, where n is the number of users in the system, in terms of concrete security. In contrast, our security bound is completely independent of the number of users.

Theorem 1. *The scheme provides* AO *security under the* DCR *assumption in the random oracle model. Namely, for any probabilistic polynomial-time adversary $\mathcal{A}$, there exists a* DCR *distinguisher $\mathcal{B}$ with comparable running time and such that*

$$\mathbf{Adv}^{\mathsf{AO}}(\mathcal{A}) \leq e \cdot (q_{enc} + 1) \cdot \mathbf{Adv}^{\mathsf{DCR}}(\mathcal{B}) \,,$$

where q_{enc} is the number of encryption queries and e is the base for the natural logarithm.

Proof. The proof proceeds with a sequence of three games. The latter begins with Game 0, which is the real game, and ends with Game 2, where even a computationally unbounded adversary has no advantage. For each $j \in \{0, 1, 2\}$, we denote by S_j the event that the challenger $\mathcal{B}$ outputs 1 in Game j. We also define $Adv_j = |\Pr[S_j] - 1/2|$.

In the sequel, we assume w.l.o.g. that the adversary $\mathcal{A}$ has always already queried the random oracle H on the input t before any encryption query for the time period t. Indeed, the challenger can always enforce this by making H-queries for itself. For simplicity, we also assume that the adversary does not query the random oracle more than once for a given t.

Game 0**:** This is the real game. Namely, the challenger performs the setup of the system by choosing $s_1, \ldots, s_n \xleftarrow{R} \pm\{0,1\}^{2\ell}$ and defining $s_0 = -\sum_{i=1}^{n} s_i$. Queries to the random oracle H are answered by returning uniformly random elements in $(\mathbb{Z}/N^2\mathbb{Z})^*$. Encryption queries $(i, t, x_{i,t})$ are answered by

returning the ciphertext $c_{i,t} = (1 + x_{i,t}N) \cdot H(t)^{s_i} \bmod N^2$. Whenever the adversary decides to corrupt some player $i \in \{0, \ldots, n\}$, the challenger reveals s_i. In the challenge phase, the adversary chooses a target time period $t^\star$, an uncorrupted subset $\mathcal{S}^\star \subseteq \mathcal{U}^\star$ and two distinct series $\langle (i, t^\star, x_{i,t^\star}^{(0)}) \rangle_{i \in \mathcal{S}^\star}$, $\langle (i, t^\star, x_{i,t^\star}^{(1)}) \rangle_{i \in \mathcal{S}^\star}$ which must satisfy Equation (1) if $\mathcal{S}^\star = \mathcal{U}^\star$ and the aggregator's private key s_0 is exposed at some point of the game (cf. § 3.1). At this stage, the challenger flips a fair binary coin $b \stackrel{R}{\leftarrow} \{0,1\}$ and the adversary $\mathcal{A}$ receives

$$\left\{ c_{i,t^\star} = (1 + x_{i,t^\star}^{(b)} N) \cdot H(t^\star)^{s_i} \bmod N^2 \right\}_{i \in \mathcal{S}^\star} .$$

We assume that the adversary queries $H(t^\star)$ before the challenge phase. Otherwise, $\mathcal{B}$ can simply make the query for itself. In the second phase, after a second series of queries, $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. We let the challenger $\mathcal{B}$ output 1 if $b' = b$ and 0 otherwise. The adversary's advantage in Game 0 is thus $Adv_0 = |\Pr[S_0] - 1/2| = \mathbf{Adv}^{\mathsf{AO}}(\mathcal{A})$.

Game 1: This game is identical to Game 0 with the following difference. For each random oracle query t, the challenger $\mathcal{B}$ flips a biased coin $\delta_t \in \{0,1\}$ that takes the value 1 with probability $1/(q_{enc}+1)$ and the value 0 with probability $q_{enc}/(q_{enc}+1)$. At the end of the game, $\mathcal{B}$ considers the event E that either of the following conditions holds:

- For the target time period $t^\star$, the coin $\delta_{t^\star}$ flipped for the hash query $H(t^\star)$ was $\delta_{t^\star} = 0$.
- There exists a time period $t \neq t^\star$ such that an encryption query $(i, t, .)$ was made for some user in $i \in \mathcal{U}^\star$ but for which $\delta_t = 1$.

If event E occurs (which $\mathcal{B}$ can detect at the end of the game), $\mathcal{B}$ halts and outputs a random bit. Otherwise, it outputs 1 if and only if $b' = b$. The same analysis as that of Coron [9] shows that $\Pr[\neg E] = 1/e(q_{enc}+1)$, where e is the base for the natural logarithm. The transition from Game 0 to Game 1 is thus a transition based on a failure event of large probability [11] and we thus have $Adv_1 = Adv_0 \cdot \Pr[\neg E] = Adv_0/e(q_{enc}+1)$.

Game 2: In this game, we modify the distribution of random oracle outputs. Specifically, the treatment of each hash query t depends on the random coin $\delta_t \in \{0,1\}$.

- If $\delta_t = 0$, the challenger $\mathcal{B}$ chooses a random N-th residue $z_t = {r_t}^N \bmod N^2$, with $r_t \stackrel{R}{\leftarrow} (\mathbb{Z}/N\mathbb{Z})^*$, and defines $H(t) = z_t$. Note that the resulting hash value $H(t)$ is now a N-th residue in $(\mathbb{Z}/N^2\mathbb{Z})^*$.
- If $\delta_t = 1$, $\mathcal{B}$ chooses a uniformly random $r_t \in (\mathbb{Z}/N^2\mathbb{Z})^*$ and programs H so as to have $H(t) = r_t$.

Lemma 1 below shows that Game 2 and Game 1 are computationally indistinguishable if the DCR assumption holds. It follows that

$$|\Pr[S_2] - \Pr[S_1]| \leq \mathbf{Adv}^{\mathsf{DCR}}(\mathcal{B}) .$$

In Game 2, we claim that $\Pr[S_2] = 1/2$ so that $\mathcal{A}$ has no advantage. Indeed, with probability $1/e(q_{enc}+1)$, we have $H(t^\star) \in_R (\mathbb{Z}/N^2\mathbb{Z})^*$ for the target time period

$t^\star$ (for which the challenge ciphertexts $\{c_{i,t^\star} = (1+N)^{x_{i,t^\star}} \cdot H(t^\star)^{s_i} \bmod N^2\}_{i\in\mathcal{S}^\star}$ are generated) whereas, for each $t \neq t^\star$, the hash value $H(t)$ is uniform in the subgroup of N-th residues. For each $t \neq t^\star$, $H(t)$ thus lives in a subgroup whose order is co-prime with N since $\gcd(N, \phi(N)) = 1$. By the Chinese Remainder Theorem, for each $i \in \mathcal{S}^\star$, the value $H(t)^{s_i}$ is completely independent of $s_i \bmod N$ when $t \neq t^\star$. In other words, encryption queries for periods $t \neq t^\star$ leak no information about $\{s_i \bmod N\}_{i\in\mathcal{S}^\star}$. At the same time, for period $t^\star$, $\{H(t^\star)^{s_i}\}_{i\in\mathcal{S}^\star}$ contain random components of order N which only appear in the challenge ciphertexts $\{c_{i,t^\star} = (1+N)^{x^{(b)}_{i,t^\star}} \cdot H(t^\star)^{s_i} \bmod N^2\}_{i\in\mathcal{S}^\star}$, where they completely blind $\{x^{(b)}_{i,t^\star}\}_{i\in\mathcal{S}^\star}$.

More precisely, let us consider what a computationally unbounded adversary $\mathcal{A}$ can see. We have the following two mutually exclusive situations.

- If $\mathcal{S}^\star \subsetneq \mathcal{U}^\star$ or $\mathcal{A}$ did not compromise the aggregator, then $\mathcal{A}$ has no information about $\{s_i \bmod N\}_{i\in\mathcal{S}^\star}$ whatsoever. Even the sum $\sum_{i\in\mathcal{S}^\star} s_i \bmod N$ remains hidden because there exists $\theta \in (\mathcal{U}^\star \backslash \{\mathcal{S}^\star\}) \cup \{0\}$ such that $s_\theta \bmod N$ is completely independent of $\mathcal{A}$'s view. If we consider individual ciphertexts $\{c_{i,t^\star}\}_{i\in\mathcal{S}^\star}$ and their partial aggregation

$$c_{\mathcal{S}^\star,t^\star} = \prod_{i\in\mathcal{S}^\star} c_{i,t^\star} \bmod N^2\,, \tag{2}$$

  for each value of $b \in \{0,1\}$, there exist $\{s_i \bmod N\}_{i\in\mathcal{S}^\star}$ that explain the adversary's view.
- If $\mathcal{S}^\star = \mathcal{U}^\star$ and $\mathcal{A}$ has obtained s_0 by corrupting the aggregator, we must have $\sum_{i\in\mathcal{S}^\star} x^{(0)}_{i,t^\star} = \sum_{i\in\mathcal{S}^\star} x^{(1)}_{i,t^\star}$. Since $\sum_{i=0}^n s_i = 0$, the only information that $\mathcal{A}$ obtains about $\{s_i \bmod N\}_{i\in\mathcal{S}^\star}$ is the sum $\sum_{i\in\mathcal{S}^\star} s_i \bmod N$, which an all-powerful adversary can infer via corruption queries or encryption queries during period $t^\star$. However, if consider the partially aggregated ciphertext (Eq. (2)), then $c^{\phi(N)}_{\mathcal{S}^\star,t^\star} \bmod N^2$ does not depend on the bit $b \in \{0,1\}$.

In Game 2, we can only have $b' = b$ with probability $1/2$. Putting all together, we therefore find

$$\mathbf{Adv}^{\mathsf{AO}}(\mathcal{A}) \leq e \cdot (q_{enc} + 1) \cdot \mathbf{Adv}^{\mathsf{DCR}}(\mathcal{B})\,,$$

which concludes the proof. □

Lemma 1. *Under the* DCR *assumption, Game* 2 *is computationally indistinguishable from Game* 1.

Proof. The proof is by contradiction and builds a DCR distinguisher $\mathcal{B}$ from an adversary $\mathcal{A}$ that has noticeably different behaviors in Game 1 and Game 2.

The reduction $\mathcal{B}$ receives as input a pair (N, z) and has to decide whether $z = r^N \bmod N^2$, for some $r \in (\mathbb{Z}/N\mathbb{Z})^*$, or $z \in_R (\mathbb{Z}/N^2\mathbb{Z})^*$. To this end, $\mathcal{B}$ begins by picking $s_1, \ldots, s_n \xleftarrow{R} \pm\{0,1\}^{2\ell}$ and sets $s_0 = -\sum_{i=1}^n s_i$, exactly as the challenger of Game 1 does. Throughout the game, $\mathcal{B}$ always answers encryption queries and

corruption queries faithfully. However, the treatment of random oracle queries $H(t)$ depends on the value of the biased coin $\delta_t \in \{0,1\}$. Namely, when $\delta_t = 0$, $\mathcal{B}$ uses the random self-reducibility of DCR and builds many instances $\{z_t\}_t$ for the same N out of z.

- If $\delta_t = 0$, $\mathcal{B}$ chooses $\alpha_t \stackrel{R}{\leftarrow} \mathbb{Z}/N\mathbb{Z}$, $\beta_t \stackrel{R}{\leftarrow} (\mathbb{Z}/N\mathbb{Z})^*$ and computes $z_t = z^{\alpha_t} \cdot {\beta_t}^N \bmod N^2$. Observe that, if z is a N-th residue (resp. a random element of $(\mathbb{Z}/N^2\mathbb{Z})^*$), z_t is uniformly distributed in the subgroup of N-th residues modulo N^2 (resp. uniformly distributed in $(\mathbb{Z}/N^2\mathbb{Z})^*$). Then, $\mathcal{B}$ programs the random oracle H to have $H(t) = z_t$.
- If $\delta_t = 1$, $\mathcal{B}$ draws $z_t \stackrel{R}{\leftarrow} (\mathbb{Z}/N^2\mathbb{Z})^*$ instead of choosing it among N-th residues. It defines $H(t) = z_t$.

When $\mathcal{A}$ terminates, $\mathcal{B}$ outputs a random bit if event E has come about during the game. Otherwise, $\mathcal{B}$ outputs 1 if $b' = b$ and 0 otherwise.

Clearly, if $z \in_R (\mathbb{Z}/N^2\mathbb{Z})^*$, $\mathcal{A}$'s view is exactly the same as in Game 1. In contrast, if z is a N-th residue, $\mathcal{B}$ is rather playing Game 2 with the adversary. □

6 Concluding Remarks

This paper presented a new scheme allowing an untrusted aggregator to evaluate the sum of user's private inputs. In contrast to prior solutions, there is no restriction on the message space or on the number of users. This results in always fast decryption and aggregation, even over large plaintext spaces and/or population of users.

Our scheme provides many other advantages over prior solutions. One of these is a much better concrete security in the random oracle model as our bound is completely independent of the number n of users. In comparison, the security proof in [25] entails a multiplicative gap proportional to n^3 in the reduction from the DDH assumption. Considering that large values of n such as $n \approx 2^{20}$ are expectable in practical applications, it seems advisable to increase the key size accordingly. In our setting, our reduction decreases the security loss to only 30 bits if we allow for $q_{enc} = 2^{30}$, as recommended in the literature [3] (note that, in this setting, this value should be seen as a rather theoretical bound since, using a counter, one can always bound q_{enc} to arbitrarily small values like, e.g., $q_{enc} \leq 3$).[2]

Finally, our scheme makes it possible to encrypt data in off-line/on-line mode through the use of coupons: once the message is known, the user only has to evaluate a single modular multiplication. It is therefore well-suited to low-power devices that are typically used in applications like smart metering or sensor networks.

[2] The proof in [25] additionally features a degradation factor of q_H, where q_H is the number of random oracle queries. We suspect that it can be reduced to q_{enc} using Coron's technique [9]. The main limiting factor in [25] is thus the cubic dependence on n.

References

1. Gergely Ács and Claude Castelluccia. I have a dream! (differentially private smart metering). In T. Filler et al., editors, *Information Hiding (IH 2011)*, volume 6958 of *Lecture Notes in Computer Science*, pages 118–132. Springer, 2011.
2. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In D. E. Denning et al., editors, *1st ACM Conference on Computer and Communications Security*, pages 399–416. ACM Press, 1993.
3. Mihir Bellare and Phillip Rogaway. The exact security of digital signatures - how to sign with RSA and Rabin. In U. M. Maurer, editor, *Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 1996.
4. Dan Boneh. The decision Diffie-Hellman problem. In J. Buhler, editor, *Algorithmic Number Theory (ANTS-III)*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.
5. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.
6. Jan Camenisch and Victor Shoup. Practical verifiable encryption and decryption of discrete logarithms. In D. Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 126–144. Springer, 2003.
7. Claude Castelluccia, Aldar C.-F. Chan, Einar Mykletun, and Gene Tsudik. Efficient and provably secure aggregation of encrypted data in wireless sensor networks. *ACM Transactions on Sensor Networks*, 5(3), 2009. Article 20.
8. T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Privacy-preserving stream aggregation with fault tolerance. In A. D. Keromytis, editor, *Financial Cryptography and Data Security (FC 2012)*, volume 7397 of *Lecture Notes in Computer Science*, pages 200–214. Springer, 2012.
9. Jean-Sébastien Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2000.
10. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In L. R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.
11. Alexander W. Dent. A note on game-hopping proofs. Cryptology ePrint Archive: Report 2006/260, 2006.
12. Cynthia Dwork. Differential privacy: A survey of results. In M. Agrawal et al., editors, *Theory and Applications of Models of Computation (TAMC 2008)*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2008.
13. Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006.
14. Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital schemes. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 263–275. Springer, 1990.
15. Flavio D. Garcia and Bart Jacobs. Privacy-friendly energy-metering via homomorphic encryption. In J. Cuéllar et al., editors, *Security and Trust Management (STM 2010)*, volume 6710 of *Lecture Notes in Computer Science*, pages 226–238. Springer, 2010.

16. Thomas Icart. How to hash into elliptic curves. In S. Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 303–316. Springer, 2009.
17. Marek Jawurek and Florian Kerschbaum. Fault-tolerant privacy-preserving statistics. In S. Fischer-Hübner and M. Wright, editors, *Privacy Enhancing Technologies (PETS 2012)*, volume 7384 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2012.
18. Marek Jawurek, Florian Kerschbaum, and George Danezis. Privacy technologies for smart grids – A survey of options. Technical Report MSR-TR-2012-119, Microsoft Research, November 2012.
19. Klaus Kursawe, George Danezis, and Markulf Kohlweiss. Privacy-friendly aggregation for the smart-grid. In S. Fischer-Hübner and N. Hopper, editors, *Privacy Enhancing Technologies (PETS 2011)*, volume 6794 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2011.
20. Fengjun Li, Bo Luo, and Peng Liu. Secure information aggregation for smart grids using homomorphic encryption. In *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2010.
21. Hsiao-Ying Lin, Wen-Guey Tzeng, Shiuan-Tzuo Shen, and Bao-Shuh Paul Lin. A practical smart metering system supporting privacy preserving billing and load monitoring. In F. Bao, editor, *Applied Cryptography and Network Security (ACNS 2012)*, volume 7341 of *Lecture Notes in Computer Science*, pages 544–560. Springer, 2012.
22. Kevin S. McCurley. A key distribution system equivalent to factoring. *Journal of Cryptology*, 1(2):95–105, 1988.
23. Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.
24. Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In A. K. Elmagarmid and D. Agrawal, editors, *2010 ACM SIGMOD International Conference on Management of Data (SIGMOD 2010)*, pages 735–746. ACM Press, 2010.
25. Elaine Shi, T.-H. Hubert Chan, Eleanor G. Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *Network and Distributed System Security Symposium (NDSS 2011)*. The Internet Society, 2011.
26. Zahava Shmuely. Composite Diffie-Hellman public key generating systems hard to break. Technical Report 356, Israel Institute of Technology, Computer Science Department, Technion, February 1985.