Privacy-Friendly Load Scheduling of Deferrable and Interruptible Domestic Appliances in Smart Grids

Cristina Rottondi and Giacomo Verticale

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci, 32, Milano, Italy
{*cristinaemma.rottondi,giacomo.verticale*}*@polimi.it*

Abstract

The massive integration of renewable energy sources in the power grid ecosystem with the aim of reducing carbon emissions must cope with their intrinsically intermittent and unpredictable nature. Therefore, the grid must improve its capability of controlling the energy demand by adapting the power consumption curve to match the trend of green energy generation. This could be done by scheduling the activities of deferrable and/or interruptible electrical appliances. However, communicating the users' needs about the usage of their appliances also leaks sensitive information about their habits and lifestyles, thus arising privacy concerns.

This paper proposes a framework to allow the coordination of energy consumption without compromising the privacy of the users: the service requests generated by the domestic appliances are divided into crypto-shares using Shamir Secret Sharing scheme and collected through an anonymous routing protocol by a set of schedulers, which schedule the requests by directly operating on the shares. We discuss the security guarantees provided by our proposed infrastructure and evaluate its performance, comparing it with the optimal scheduling obtained by means of an Integer Linear Programming formulation.

Keywords: Load Scheduling, Shamir Secret Sharing, Smart Grid

[☆] A preliminary version appeared in C.Rottondi and G. Verticale, Privacy-Friendly Appliance Load Scheduling in Smart Grids, *SmartGridComm 2013, IEEE International Conference on Smart Grid Communications*, Vancouver, Canada, October 2013

2010 MSC: 00-01, 99-00

1. Introduction

One of the most relevant goals in the design of the future energy grid is the massive introduction of power plants exploiting Renewable Energy Sources (RES, e.g. wind, solar and geothermal energy) to reduce carbon emission and shift towards a more sustainable power usage. However, due to the intrinsic unpredictability in the production of "green" power caused by the intermittent nature of renewables, the new Smart Grid scenario will cope with numerous issues related to the balancing of energy generation and consumption within the grid, in order to satisfy the energy demand while avoiding power waste. In addition, the energy market will experience more uncertain conditions, which could possibly affect the dynamics of energy pricing [1].

In order to increase the flexibility of the energy utilization, three complementary approaches have been proposed. The first is to equip the grid with high capacity storage banks, capable of storing energy surpluses and to release them in case of energy production deficits [2]. However, today's state-of-the-art technology is still immature to allow a widespread introduction of storage plants, which would require tremendous installation and maintenance costs. A second possibility is to induce some modifications in the user's energy utilization behavior by designing time-variable tariffs or introducing incentives to shift the use of some appliances to off-peak hours [3]. Unfortunately, this approach does not provide any form of direct control on the load conditions of the grid. Finally, the third alternative relies on load scheduling approaches operating at single household level or at neighborhood/microgrid level with the aim of shaping the energy demand profile in order to meet the production trend. Such mechanisms work according to the following principle: delay-tolerant and/or interruptible operations can be scheduled and initiated only when the green energy production conditions are favorable, while in case of power shortage the starting time can be postponed or the service can be momentarily interrupted. Moreover, a

wide category of appliances (e.g. refrigerators, air conditioning, cooling/heating systems) can tune (up to a certain extent) their power consumption according to the grid state.

The drawback of the load scheduling approach is that it requires the users to communicate to the scheduler their preferences about the time of use and the energy consumption profile of the appliances to be scheduled, which makes the system prone to appliance load monitoring attacks (e.g. Non Intrusive Load Monitoring (NILM), load disaggregation algorithms, and transient analysis). In fact, it has been widely proved that, by analyzing the power consumption trend of an individual household, very detailed information about the personal habits of the occupants can be inferred [4, 5], making it possible even to identify the specific electrical appliances working in a given time period. Therefore, designing a load scheduling system capable of preserving the privacy of the users is still an open issue.

In this paper, we propose a privacy-friendly infrastructure to perform appliance load scheduling within a neighborhood, which directly exposes neither the time of use and the energy consumption pattern of the single appliances, nor the identity of the users specifying the scheduling requests. Our solution relies on a set of schedulers which collaboratively perform the load planning by means of a MultiParty Computation (MPC) protocol based on Shamir Secret Sharing scheme. The proposed architecture is in line with the recent proposals by regulation bodies: for example, the California Public Utilities Commission [6] fosters the realization of Energy Data Centers aimed at the collection and elaboration of energy consumption data and run by governmental or public entities. While such Data Centers are assumed to be honest, our proposed architecture ensures no violation of the customers' privacy even in presence of collectors behaving according to the "honest but curious" model.

The remainder of the paper is structured as follows: Section 2 provides a short overview of the related literature, while Section 3 recalls some background notions. Section 4 describes the privacy-preserving scheduling architecture. The attacker model and the security analysis of our proposed infrastructure are

discussed in Section 5. In Section 6, the scheduling problem is formulated as an Integer Linear Program (ILP), which is used as a benchmark for the complexity and performance assessment provided in Section 7. Conclusions are drawn in the final Section.

2. Related Work

Various models for energy load management systems have been recently proposed by the research community: in [7], an optimal and automatic residential energy consumption scheduling framework is described, which attempts to strike a balance between minimizing the electricity payment and minimizing the waiting time for the operation of each appliance in the household, in presence of time-variable tariffs. The problem is modelled by means of a linear programming formulation and a weighted average price prediction filter is used to estimate the future trend of the energy tariff. A real-time residential load management model and algorithm is also discussed in [8], which differentiates the scheduling policy according to the type of electrical appliances to be served (interruptible, non interruptible and must-run). However, in both cases the system is designed for a single household, while our scheduling framework is aimed at controlling multiple residential buildings. The authors of [9] propose a neighborhood scheduler that divides the energy requests in queues according to their shape and priority and optimizes the service time of deferrable individual appliances (e.g. washing machines, dishwashers, cloth dryers, and electric vehicles recharge). In case the electrical appliances are assumed to have rectangular energy consumption profile, the scheduling problem can be treated as a rectangle/strip packing problem, which has been thoroughly investigated, for example in [10, 11, 12], and consists in optimally placing a set of rectangles of different dimensions in a two-dimensional space of given width and infinite height.

In our framework, differently from [7], the optimization goal is to shape the cumulative energy consumption of a set of appliances according to the availability of energy generated by renewable energy sources. We deal with the

same scenario and appliance category of [9], but with respect to [9] we also consider must-run and interruptible appliances. Conversely, [12] proposes an online power strip packing algorithm for malleable energy demands with rectangular shape, providing performance guarantees in terms of upper bounds with respect to the optimal solution. Apart from the different appliance category, though our solution does not provide any guarantee on the quality of service experienced by the users, it deals with appliances having a generic energy consumption curve.

Though the problem of securely managing the energy consumption data has been widely studied in the context of the Automatic Metering Infrastructure (AMI) of Smart Grid, to the best of our knowledge this is the one of the first studies specifically dealing with data security in a load scheduling framework. Moreover, the security assumptions modeling the adversarial entities which attempt to access users' data are quite various and most often too loose with respect to realistic attack scenarios. Paper [13] proposes a distributed architecture in which multiple Home Energy Management (HEM) units collaborate with each other in order to keep the demand and supply balanced in their neighbourhood by solving a multi-stage stochastic optimization problem. The proposed system hides the users' individual information to any external entity (e.g., energy provider or grid manager) but requires the customers to communicate their power schedules to their neighbours. Conversely, paper [14] avoids data exchange among households, but assumes a trusted energy utility to collect the individual power consumption curves and to broadcast price information which are updated at every game iteration. Our solution does not require any end-to-end communication among the households: neighboring meters acts exclusively as relays running the Crowds protocol to transmit messages which are ciphered by means of an hybrid encryption algorithm. Furthermore, our framework assumes a pool of possibly colluding honest-but-curious schedulers, ensuring that no information leaks occurs in case at least one scheduler is not colluded. Papers [15, 16] assume that exchanging aggregated power consumption data at household level (e.g., on hourly basis) is sufficient to hide the usage patterns of single electric appliances to untrustworthy neighbours. However,

several studies on Non-Intrusive Load Monitoring (see, e.g., [17, 18]) prove that the power consumption patterns of individual appliances can easily be inferred from house-aggregated measurements. To defy such kind of attacks, our proposed protocol hides the energy consumption patterns of individual appliances to both schedulers and neighboring meters by means of Shamir Secret Sharing scheme: the only information disclosed to the schedulers is the appliance category (must-run, deferrable, interruptible) and the feasible starting times.

Among the techniques proposed to securely collect meter readings, paper [19] describes a wavelet-based data perturbation method to allow multiple entities to access the data generated by a meter with different levels of detail, according to their needs and access rights. Alternative techniques rely on data perturbation [20, 21, 22], pseudonymization [23, 24, 25], or on data aggregation by means of MultiParty Computation [26, 27, 28]. Our proposed privacy-preserving scheduling infrastructure is inspired by the one originally presented in [29] and adapted to the Smart Grid context in [28], and is based on the same homomorphic encryption scheme, named Shamir Secret Sharing. However, with respect to [28], which deals with the secure collection of meter readings in AMI, in this paper we cope with an inherently different problem, characterized by peculiar privacy requirements that must be addressed by means of specifically designed security solutions.

3. Background

3.1. Shamir Secret Sharing Scheme

Shamir Secret Sharing (SSS) scheme [30] belongs to the family of cryptographic threshold schemes, which are designed to allow the collaborative reconstruction of a secret. In a (w, t)-threshold scheme, the secret is divided in w parts called *shares*, which are distributed among the protocol participants and can be reconstructed if at least $t \leq w$ participants cooperate.

The SSS scheme works as follows: let $m \in Z_q$ be the secret, where q is a prime number, greater than w and than all the possible secrets. To split

the secret in w shares, chose $t-1$ integer random numbers $\rho_1, \rho_2, \cdots, \rho_{t-1}$ with uniform distribution in $[0, q-1]$ and calculate the s-th share of the secret m, (x_s, y_s) for $1 \leq s \leq w$, where x_s are distinct integer numbers and $y_s = m + \rho_1 x_s + \rho_2 x_s^2 + \ldots + \rho_{t-1} x_s^{t-1} \bmod q$. The secret can be recovered in presence of at least t shares by using the Lagrange interpolation method. The SSS scheme is fully homomorphic, meaning that both addition and multiplication can be performed directly on the shares, leading to the same result that would be obtained by computing the same operation on the plaintext. More in detail, the sum of two secrets can be locally computed by each participant by summing the corresponding shares. Conversely, multiplication requires a collaborative procedure, such as the one described in [31]. Therefore, any function that can be expressed in terms of additions and multiplications can be computed by operating on the shares, albeit with different complexity. In particular, several collaborative methods to perform the comparison of two secrets have been proposed (see e.g. [32, 33]). In the remainder of the paper, we will adopt the procedure first introduced in [34] and described in [33], which works as follows: each party holding the s-th shares $(x_s, y_s), (x'_s, y'_s)$ of the secrets m and m' to be compared selects two big random numbers r_s, r'_s, which can multiplicatively hide $m - m'$, and a random bit $b_s \in \{0, 1\}$. The collaborative protocol enables each party to obtain a share of the quantity $c = (m - m') \prod_{s=1}^{t} (-1)^{b_s} r_s - \sum_{s=1}^{t} (-1)^{1-b_s} r'_s$. The result of the comparison can be computed by retrieving c, setting a bit e either to 0 in case $c > 0$ or to 1 otherwise[1], and calculating the result of the XOR operation $\xi = e \oplus b_1 \oplus \cdots \oplus b_t$. $\xi = 0$ indicates that $m > m'$, while $\xi = 1$ indicates that $m \leq m'$. The reader is referred to [33] for additional details about the collaborative procedure and the proof of the correctness of the comparison protocol.

[1] Note that in a modulo n field negative numbers are represented by the upper half of the range $[0, n-1]$.

3.2. Anonymous Routing with Crowds

Crowds is an anonymous routing protocol originally proposed in [35] to hide the true sender of a message by routing it randomly within a large group of users (the *crowd*). The protocol assumes the presence of a central node called *blender*, which is responsible of providing each node with the list of active crowd members and of updating it periodically, and that the communications between any two members of the crowd are encrypted using a symmetric key encryption scheme. Upon receipt of a message, each crowd member P behaves as follows: with probability $p > 0.5$, it forwards the message to a randomly chosen node within the crowd (possibly itself), otherwise it sends the node to the final addressee (see Algorithm 1). The most important anonymization property of Crowds is that the entity to which the messages are sent is equally likely to receive the message from any crowd member independently of the original sender, i.e.

$$P(\tilde{G} = x | G = g) = P(\tilde{G} = x) \tag{1}$$

where $\tilde{G}$ is the random variable indicating the last hop of the message and G is the random variable indicating the original source of the message. For a proof of this statement and a detailed security analysis of the protocol, the reader is referred to [35].

It is worth noting that our privacy-preservation protocol can be built upon other anonymous routing protocols, provided that they guarantee security properties similar to (1). We think Crowds is particularly well suited for the Neighborhood Area Networks of Smart Grids, which may comprise only a limited number of nodes, because its security properties do not depend on the size of the network.

4. The Privacy-Friendly Load Scheduling Framework

As depicted in Fig. 1, our proposed architecture comprises a set of Appliances, $\mathcal{A}$, each one generating its own load scheduling requests, and a set of Schedulers, $\mathcal{S}$, which collaboratively define the starting delay of the service requests received from the Appliances. Note that, as in [9], we consider deferrable

Algorithm 1 Crowds routine executed by each crowd member P

if P is the sender of message M **and** M has never been forwarded **then**
  Select a uniformly random node within the crowd and forward M to it
else
  Receive the pair (Node P', Message M)
  Flip a biased coin such that ($Pr(Heads) = p$)
  if $Heads$ **then**
    Select a uniformly random node within the crowd and forward M to it
  else
    Forward M to destination
  end if
end if

Appliances without providing any guarantee on the maximum delay imposed by the scheduling algorithm on their starting times. However, with respect to [9] we also include must-run and interruptible Appliances in our analysis: service requests generated by must-run devices must be served upon arrival without delays nor interruptions, while interruptible appliances tolerate not only an initial service delay but also intermediate interruptions.

The architecture also includes a Smart Gateway in each household, which is equipped with secure communication capabilities (e.g. as the one proposed by the German Federal Office for Information Security in [36]) and is responsible of gathering the service requests generated by the Appliances inside the building and to convey them to the Schedulers. In the following we will indicate as $\mathcal{G}$ the set of Gateways. We also assume that:

1. The parties agree on a hybrid encryption algorithm $E(K_e, \cdot)$ and a corresponding decryption algorithm $D(K_d, \cdot)$. The hybrid scheme is assumed to be IND-CPA secure for equally sized messages [37] (i.e. it ensures message indistinguishability under chosen plaintext attack) and uses state-of-the art secure public key and symmetric cryptography to transmit messages of any size.

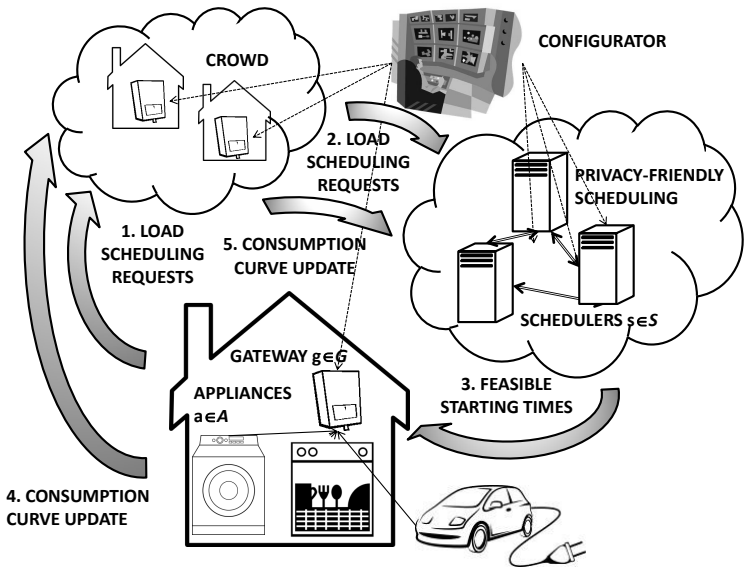

Figure 1: The privacy-friendly load scheduling infrastructure

2. Each Scheduler $s \in \mathcal{S}$ $(1 \leq s \leq w)$ has its own pair of public/private keys (K_e^s, K_d^s) and all the Gateways know the public keys of the Schedulers.
3. All the communication channels among the nodes of the architecture are confidential and authenticated hop-by-hop.
4. A Configurator node acts as a blender for the Crowds routing protocol.
5. The scheduling horizon is divided in $\tilde{T}$ time slots.

The design goal is to make the Schedulers define the starting times of the load scheduling requests generated by the Appliances without learning their energy consumption curve nor their owner. In order to hide the energy consumption pattern of the devices, the Gateways generate multiple delayed versions of the energy consumption curve of the local Appliances (possibly interleaved by interruptions), and encrypt each of them with the SSS scheme. The shares obtained from such data must be anonymously distributed among the Schedulers in a randomized order: to do so, each set of shares (one for each sample of the curve) is conveyed to the respective Scheduler by means of the anonymous routing protocol Crowds.

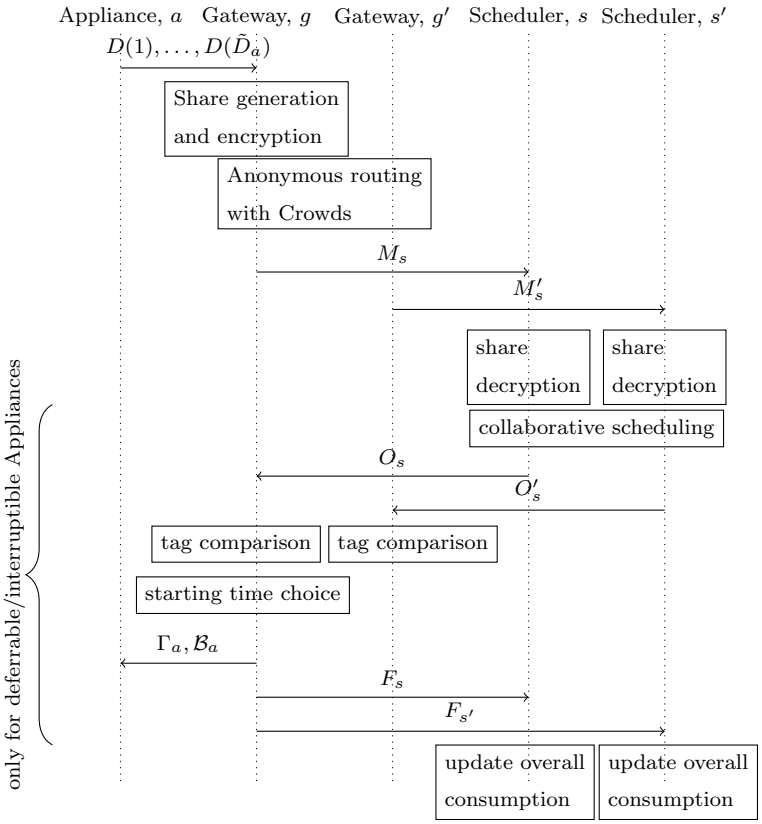

Figure 2: Data exchange during the load scheduling procedure. Message M_s is defined by either Eq. (2), (3), or (4). Message O_s is defined by either Eq. (5) or (7). message F_s is defined by Eq. (6).

Once all the Schedulers have received their shares, they verify the feasibility of each delayed curve by checking whether the total power load of the Appliances already scheduled does not exceed the expected amount of energy generated by the RESes. We assume that such expected supply curve $T(i)$ $(i \leq 1 \leq \tilde{T})$ is public and known to all the Schedulers. Such procedure is performed by means of a collaborative protocol based on the one proposed in [33], which enables the Schedulers to make comparisons by operating directly on the shares. Then, the Schedulers communicate to the local Gateway the feasibility of each delayed curve. The protocol guarantees that the Schedulers know neither the

shape or shift of each consumption curve, nor whether, for a given shift, the corresponding shifted curve is feasible. The local Gateway schedules the lowest feasible starting time (and possible intermediate interruptions) and instructs the Appliance accordingly. Finally, the shares of the shifted curve corresponding to the delay scheduled by the Gateway are sent to the Schedulers, which use them to update their information about the overall energy consumption within the neighborhood by operating again directly on the shares.

Figure 2 provides a pictorial view of the messages exchanged during execution of the protocol.

Let $D_a(i)$ be a sequence of samples of the load profile curve of the Appliance $a \in \mathcal{A}$ (with $1 \leq i \leq \tilde{D}_a$, where $\tilde{D}_a$ is the sequence length). The sampling rate is set to one sample per slot. If a initiates a new service request at slot τ ($1 \leq \tau < \tilde{T} - \tilde{D}_a$), it sends $D_a(i)$ to the local Gateway g. Let Γ_a be the chosen starting time of a and $\mathcal{B}_a$ be the set of the chosen intermediate interruptions while serving a. According to the appliance type AT_a (must-run, deferrable, or deferrable and interruptible), the Gateway g operates as follows:

- if a is a must-run Appliance, g computes a sequence $V(i)_a$ of length $\tilde{T}$, where:

$$V(i)_a = \begin{cases} D_a(i-\tau) & \text{if } \tau < i \leq \tau + \tilde{D}_a \\ 0 & \text{otherwise} \end{cases}$$

  which corresponds to setting the Appliance starting time Γ_a to the following time slot (i.e. $\Gamma_a = \tau + 1$), without any intermediate interruption (i.e. $\mathcal{B}_a = \oslash$). Each sample $V_a(i)$ is divided in w shares $S_s^{a,i} = (x_s, y_s^{a,i})$, where x_s is the ID of Scheduler s. Note that the random coefficients $\rho_1, \rho_2, \cdots, \rho_{t-1}$ used by the SSS scheme are refreshed for each sample. Then, g generates an ephemeral encryption/decryption keypair (K_e^g, K_d^g) for the hybrid encryption scheme, a nonce r_a, which is used as a tag associated to the service request generated by the ath Appliance, and sends the message:

$$M_s = E(K_e^s, K_e^g \| AT_a \| r_a \| S_s^{a,1} \| \ldots \| S_s^{a,\tilde{T}}) \tag{2}$$

to the sth Scheduler[2] by means of the anonymous routing protocol Crowds with forwarding probability p.

- if a is a deferrable Appliance, g computes $\tilde{T}-\tau$ sequences $V(i)_a^j$ $(0 \leq j \leq \tilde{T}-\tau-1)$ of length $\tilde{T}$, where:

$$V(i)_a^j = \begin{cases} D_a(i-j-\tau) & \text{if } \tau+j < i \leq \tau+j+\tilde{D}_a \\ 0 & \text{otherwise} \end{cases}$$

  Such sequences correspond to different starting times, ranging from $\Gamma_a = \tau+1$ to $\Gamma_a = \tilde{T}-\tilde{D}_a$, without any intermediate interruption (i.e. $\mathcal{B}_a = \oslash$). Then, g divides every sample of each sequence in w shares using SSS scheme, thus obtaining $\tilde{T}(\tilde{T}-\tau)$ sets of w shares $S_s^{a,i,j}$, generates (K_e^g, K_d^g) and r_a, and sends to the sth Scheduler the message:

$$M_s = E(K_e^s, K_e^g \| AT_a \| r_a \| \mathcal{P}_j(S_s^{a,1,j} \| \ldots \| S_s^{a,\tilde{T},j})) \tag{3}$$

  where $\mathcal{P}_j$ indicates a random permutation of the sequences $S_s^{a,1,j} \| \ldots \| S_s^{a,\tilde{T},j}$ over the possible values j of the shift.

- if a is a deferrable and interruptible Appliance, g computes $(\tilde{T}-\tau)^2$ sequences $V(i)_a^{j,z}$ $(0 \leq j \leq \tilde{T}-\tau-1,\ 0 \leq z \leq \tilde{T}-\tau-1)$ of length $\tilde{T}$, where:

$$V(i)_a^{j,z} = \begin{cases} D_a(i-j-\tau) & \text{if } \tau+j < i \leq \tau+j+\tilde{D}_a \wedge i-j-\tau = z+1 \\ 0 & \text{otherwise} \end{cases}$$

  Such sequences correspond to all the possible delays experienced by each single sample of the energy consumption curve of a, within the scheduling time span. Note that each sample is delayed regardless to the delays of the preceding and successive samples, since excluding the vectors which

[2]For the sake of easiness, in this paper we set as SSS threshold $t = w$, meaning that all the Schedulers must collaborate to perform the scheduling procedure. However, to improve resiliency to faults and malfunctions, t could be lower than w. For a discussion on the correct dimensioning of t and w to improve resiliency, the reader is referred to [28].

do not respect the correct sample ordering would impact on the length of M_s, which may leak information about the curve length $\tilde{D}_a$. Then, g generates $\tilde{T}(\tilde{T}-\tau)^2$ sets of w shares $S_s^{a,i,j,z}$ (w shares for each sample of the $(\tilde{T}-\tau)^2$ sequences of length $\tilde{T}$), (K_e^g, K_d^g) and r_a, and sends to the sth Scheduler the message:

$$M_s = E(K_e^s, K_e^g \| AT_a \| r_a \| \mathcal{P}_{j,z}(S_s^{a,1,j,z} \| \dots \| S_s^{a,\tilde{T},j,z})) \tag{4}$$

The expected daily energy production by RESes is expressed by the sequence $T(i)$ $(1 \leq i \leq \tilde{T})$, which is known to each Scheduler. Moreover, every Scheduler locally stores the shares of a sequence $P(i)$ which records the overall power load experienced by the grid, computed as the sum of the energy consumption curves of all the appliances already scheduled. Such shares are initialized as $P_s(i) = 0$ for $1 \leq i \leq \tilde{T}, 1 \leq s \leq w$ at the beginning of the scheduling horizon.

Upon reception of a new message M_s at slot τ, the s-th Scheduler operates as follows:

1 It decrypts M_s using its decryption key K_d^s and reads the application type AT_a.

2.a If a is a must-run Appliance, it assumes that the starting time of a is scheduled for the slot $\tau+1$, and updates $P_s(i) = P_s(i) + y_s^{a,i}$ for $1 \leq i \leq \tilde{T}$, regardless to the value of $T(i)$. Note that, thanks to the homomorphic properties of SSS with respect to addition, increasing the actual load curve with the contribution of the new appliance can be done by operating directly on the shares.

2.b If a is a deferrable (but non-interruptible) Appliance, for every sample i of each sequence j, it computes $P'_s(i) = P_s(i) + y_s^{a,i,j}$, collaboratively compares $P'(i)$ to $T(i)$ with the other Schedulers according to the protocol defined in [33], and stores the sth share $S_s^{i,j}(c)$ of the comparison output $c^{i,j}$ and the associated random bit $b_s^{i,j}$. Then, it generates the message:

$$O_s = r_a \| E(K_e^g, S_s^{a,1} \| \dots \| \mathcal{P}_j(S_s^{1,j}(c) \| b_s^{1,j} \| \dots \| S_s^{\tilde{T},j}(c) \| b_s^{\tilde{T},j})) \tag{5}$$

and broadcasts it to every Gateway $g \in \mathcal{G}$ (note that broadcasting the message is necessary, since the identity of the sender of the load request is unknown to the Schedulers due to the usage of the Crowds protocol). Upon reception of the w messages O_s (one from each of the w Schedulers), each Gateway compares the tags associated to the requests generated by the local Appliances to r_a. In case of matching, for every sample i of each sequence j, it recovers $c^{i,j}$ by means of the w collected shares $S_s^{i,j}(c)$ and computes the final comparison result $\xi^{i,j}$ according to [33]. Then, for each sequence j, g calculates $\Xi^j = \xi^{1,j} \wedge \ldots \wedge \xi^{\tilde{T},j}$. Finally, g schedules $\Gamma_a = \tau + 1 + \overline{j}$, where $\overline{j} = \min_{\{j\colon \Xi^j = 1\}} j$, communicates the pair $\Gamma_a, \mathcal{B}_a$ to a, and anonymously sends to each Scheduler the shares of the $\overline{j}$th sequence by means of the Crowds protocol:

$$F_s = E(K_e^s, S_s^{a,1,\overline{j}} \| \ldots \| S_s^{a,\tilde{T},\overline{j}}) \tag{6}$$

In turn, each Scheduler s replaces $P_s(i)$ with $P_s(i) + y_s^{a,i,\overline{j}}$ $(1 \leq i \leq \tilde{T})$. In case $\{j\colon \Xi^j = 1\} = \oslash$, an error message is returned and the local household must decide whether to serve the Appliance with non RES energy or not to run the Appliance at all. A discussion on the service policies to be applied to ensure fairness in case of RES-energy shortage is left for future investigation.

2.c If a is a deferrable and interruptible Appliance, the same procedure described in 2.b is performed for every sample i of each sequence identified by the pair (j, z). Then, each Scheduler s broadcasts the message:

$$O_s = r_a \| E(K_e^g, \mathcal{P}_{j,z}(S_s^{1,j,z}(c) \| b_s^{1,j,z} \| \ldots \| S_s^{\tilde{T},j,z}(c) \| b_s^{\tilde{T},j,z})) \tag{7}$$

The Gateway locally connected to the Appliance a obtains the final comparison result $\xi^{i,j,z}$ for each of the $\tilde{T}(\tilde{T} - \tau)^2$ samples, then for each sequence it calculates $\Xi^{j,z} = \xi^{1,j,z} \wedge \ldots \wedge \xi^{\tilde{T},j,z}$. Finally, it computes Γ_a and $\mathcal{B}_a$ according to Algorithm 2, conveys them to a and anonymously forwards the algorithm output message F_s to the Schedulers, which update $P(i)$ as described at step 2.b.

Algorithm 2 Computation of starting time and intermediate interruptions of deferrable and interruptible Appliances

Initialize $\Gamma_a \leftarrow 0, \mathcal{B} \leftarrow \oslash$

Initialize $\overline{j}(z) \leftarrow 0 \ \forall z\colon 0 \leq z \leq \tilde{D}_a - 1$

Initialize $\overline{y}_s^{a,i} \leftarrow 0 \ \forall s\colon 1 \leq s \leq w, i\colon 1 \leq i \leq \tilde{T}$

for all $z = 0$ to $\tilde{D}_a - 1$ **do**

  $\mathcal{J} \leftarrow \{j\colon \Xi^{j,z} = 1 \wedge j > \overline{j}(z-1)\}$

  if $\mathcal{J} = \oslash$ **then**

    return warning message

  else

    $\overline{j}(z) \leftarrow \min_{j \in \mathcal{J}} j$

    if $\overline{j}(z) > \overline{j}(z-1) + 1$ **then**

      $\mathcal{B}_a \leftarrow \mathcal{B}_a \cup \{\overline{j}(z-1) + 1, \cdots, \overline{j}(z) - 1\}$

    end if

  end if

  $\Gamma_a \leftarrow \overline{j}(0)$

  for all $s = 1 : w, i = 1 : \tilde{T}$ **do**

    $\overline{y}_s^{a,i} \leftarrow \overline{y}_s^{a,i} + y_s^{a,i,\overline{j},z}$

  end for

end for

return $F_s = (K_e^s, (x_s, \overline{y}_s^{a,1}), \cdots, (x_s, \overline{y}_s^{a,\tilde{T}})) \ \forall s\colon 1 \leq s \leq w$

For the sake of easiness, we do not discuss the case of multiple requests arriving in a short time interval: we assume that the Schedulers are able to process multiple requests without ambiguities.

5. Attacker Model and Security Analysis

5.1. Attacker Model

We assume a scenario where both Gateways and Schedulers behave according to the *honest-but-curious* attacker model: they obey to the protocol rules but try to infer the identities of the owners of active electrical appliances and the type of appliance being used. The first objective can be achieved by associating the service requests to the identifier of the Gateway initiating them e.g. through a linking attack, while the second implies the application of appliance load monitoring techniques. Conversely, we assume that the time of use of the appliances does not represent, by itself, a sensitive information, as long as it cannot be linked to the owner nor to the type of the electrical appliance. However, it is worth noting that the probability of success of such kind of linking attacks, which fall into the field of traffic analysis, decreases when the number of protocol participants increase, and can therefore be lowered by properly setting the size of the set $\mathcal{A}$. A possible countermeasure to linking attacks is to introduce random scheduling delays for each appliance to be scheduled. Unfortunately, such approach would lead to a strong degradation of the protocol performance in terms of the average delay experienced by the users, which would be intolerable in real scenarios. Moreover, the discussion of the impact of timing attacks on both the Crowds routing and scheduling protocol is out of the scope of this paper.

Though in this paper we do not discuss the case of dishonest users, it is worth noting that the protocol discourages selfish users from declaring their appliances to be must-run, regardless to their real type, in order to eliminate their experienced scheduling delay: since the protocol aims at adapting the overall power load to the energy production by RESes, a large fraction of must-

run appliances would greatly reduce the flexibility of the aggregate load. In case of economic incentives for the usage of renewables (e.g. lower energy price), a schedule which does not take advantage of the availability of RESes would in turn cause an increase in the energy price, thus affecting the energy bill of all the users (including the cheaters).

Moreover, a wide category of intrusive attacks aimed at Denial of Service (DoS), can be mitigated by imposing a threshold on the maximum number of daily scheduling requests (e.g. a few tens per day), which avoids the generation of fake service requests by dishonest Gateways. Such fake requests would inevitably increase the delay experienced by the Appliances run by honest users. Conversely, the effect of malicious Gateways deviating from the standard Crowds message forwarding routine (e.g. dropping messages instead of relaying them, thus preventing the requests generated by honest users from being processed), can be alleviated by lowering the SSS threshold t, which ensures the correct execution of collaborative comparison procedure even in case of up to $w - t$ missing shares.

Obliviousness. Similarly to [28], we define the architecture as **oblivious** if a collusion of any number of Gateways cannot obtain information about the power consumption pattern of the scheduled electrical appliances of the same type, except for the ones belonging to the local household. More formally, we define the `Obliv` experiment, which involves a challenger $\mathcal{C}$ and a probabilistic polynomial-time adversary $\mathcal{D}$ controlling the whole set of Gateways:

1. At a given time slot τ $\mathcal{D}$ selects two Appliances $a_0, a_1 \in \mathcal{A}$ and communicates to $\mathcal{C}$ the appliance types AT_{a_0}, AT_{a_1}, the tags r_{a_0}, r_{a_1}, the Gateway encryption key K_e^g the consumption profiles $V_{a_0}(i), V_{a_1}(i)$ for $1 \leq i \leq \tilde{T}$, and the random numbers $\rho_1, \rho_2, \ldots, \rho_{t-1}$ to be used to divide each sample of $V_{a_0}(i)$ and $V_{a_1}(i)$ in shares.
2. $\mathcal{C}$ selects a random bit $b = \{0, 1\}$, generates $\mathcal{V}_b = \{E(K_e^s, K_e^g \| AT_{a_b} \| r_{a_b} \| S_s^{a_b,1} \| \ldots \| S_s^{a_b,\tilde{T}})\ \forall s \in \mathcal{S}\}$ and communicates it to $\mathcal{D}$.
3. $\mathcal{D}$ outputs a bit b'.

The architecture provides **obliviousness** if:

$$P(b' = b \mid \mathcal{V}_b) = P(b' = b) = \frac{1}{2}$$

Blindness. Moreover, we say that the architecture is t-**blind** if a collusion of less than t Schedulers cannot learn anything about the energy consumption trend of the appliances to be scheduled, except their type. To formalize this property, we define the `Blind` experiment, involving a challenger $\mathcal{C}$ and an adversary $\mathcal{D}$ controlling a set of Schedulers $\tilde{\mathcal{S}}$: $|\tilde{\mathcal{S}}| < t$:

1. At a given time slot τ, $\mathcal{D}$ selects two Appliances $a_0, a_1 \in \mathcal{A}$ and the energy production profile by RESes $T(i)$, and communicates $V_{a_0}(i), V_{a_1}(i), T(i)$ for $1 \leq i \leq \tilde{T}$ to $\mathcal{C}$, together with an arbitrarily chosen overall energy consumption curve of the Appliances already scheduled.
2. $\mathcal{C}$ selects a random bit $\overline{b} = \{0, 1\}$ and generates $\mathcal{V}_{\overline{b}} = \{(S_s^{a_{\overline{b}},1}, \ldots, S_s^{a_{\overline{b}},\tilde{T}}) \; \forall s \in \tilde{\mathcal{S}}\}$. Moreover, $\mathcal{C}$ generates the shares $P_s(i) \; \forall s \in \tilde{\mathcal{S}}$ of the overall energy consumption curve provided by $\mathcal{D}$, runs the comparison protocol described in [33], stores a list $I_s^{a_{\overline{b}}}$ of the messages received/sent by each Scheduler $s \in \tilde{\mathcal{S}}$ during the protocol execution, obtains the comparison outputs $\mathcal{O}_{\overline{b}} = \{S_s^{a_{\overline{b}},1}(c), \ldots, S_s^{a_{\overline{b}},\tilde{T}}(c), b_s^{a_{\overline{b}},1}, \ldots, b_s^{a_{\overline{b}},\tilde{T}} \; \forall s \in \tilde{\mathcal{S}}\}$, and communicates $\mathcal{V}_{\overline{b}}, \mathcal{I}_{\overline{b}} = \{I_s^{a_{\overline{b}}} \; \forall s \in \tilde{\mathcal{S}}\}, \mathcal{O}_{\overline{b}}$ to $\mathcal{D}$.
3. $\mathcal{D}$ outputs a bit $\overline{b}'$.

The architecture provides t-**blindness** if:

$$P(\overline{b}' = \overline{b} \mid \mathcal{V}_{\overline{b}}, \mathcal{I}_{\overline{b}}, \mathcal{O}_{\overline{b}}) = P(\overline{b}' = \overline{b}) = \frac{1}{2}$$

Sender Anonymity. Finally, according to the definition in [38], the architecture provides **sender anonymity** if a collusion of any number of Schedulers cannot associate a request to the identity of the user whose Appliance generated it. Adopting the same formalization of [35], we define the following experiment, named `S-Anon`, which assumes a challenger $\mathcal{C}$ and an adversary $\mathcal{D}$ controlling the whole set of Schedulers $\mathcal{S}$:

1. $\mathcal{D}$ selects an Appliance $a \in \mathcal{A}$ and communicates $V_a(i)$ for $1 \leq i \leq \tilde{T}$ to $\mathcal{C}$.

2. $\mathcal{C}$ generates the set of encrypted measurements $\mathcal{V} = \{E(K_e^s, K_e^g \| AT_a \| r_a \| S_s^{a,1} \| \ldots \| S_s^{a,\tilde{T}}) \; \forall s \in \mathcal{S}\}$ and forwards each of them to the addressed Scheduler s by means of the Crowds routine until they reach the addressee.
3. $\mathcal{D}$ outputs the guess g' of the identifier g of the Gateway locally connected to a.

The architecture provides **sender anonymity** if:

$$P(g = g') \leq \frac{1}{|\mathcal{G}|}$$

5.2. Security Analysis

We now discuss how the security properties defined in Section 5.1 are satisfied by our proposed infrastructure, also providing formal proofs.

Theorem 1. *Under the assumption that the cryptosystem $E(K_e, \cdot)$ ensures message indistinguishability, the privacy-preserving load scheduling infrastructure provides **obliviousness** for Appliances of the same type.*

Proof. By contradiction, let us assume that the adversary $\mathcal{D}$ has more than negligible advantage in the `Obliv` experiment. Since in `Obliv` the adversary $\mathcal{A}$ arbitrarily chooses all the parameters to construct the plaintext messages $K_e^g \| AT_{a_b} \| r_{a_b} \| S_s^{a_b,1} \| \ldots \| S_s^{a_b,\tilde{V}} \forall s \in \mathcal{S}$ and the messages are equally sized if the two Appliances are of the same type, the `Obliv` is constructed analogously to the IND-CPA experiment [37]. It follows that, if $\mathcal{D}$ has more than negligible advantage over randomness to guess b in the `Obliv` experiment, it also has a non-negligible advantage in the IND-CPA experiment, which violates the assumption of message indistinguishability under chosen plaintext. Therefore, even if a collusion of Gateways collects all the w encrypted measurements of a given service request, it cannot obtain any information on the encrypted data. □

Theorem 2. *Assuming the usage of a (w, t)-SSS scheme ($t \leq w$), the privacy-preserving load scheduling infrastructure provides t-**blindness** for Appliances of the same type.*

Proof. The proof is a consequence of the property of *perfect secrecy* of the SSS scheme [39] and shows that the contents of all the input/output messages received/sent by the collusion $\tilde{\mathcal{S}}$ during the scheduling procedure leak no information about $\overline{b}$. For the single SSS-encrypted ith sample, the proof is similar to the one provided in [40, Theorem 3]: let A_0, A_1 be the random variables indicating the ith sample of Appliances a_0, a_1. Since the value of A_0 is completely determined by knowledge of A_1, it follows that:

$$P(\overline{b} = 0 \mid S_s^{a_{\overline{b}},i}, I_s^{a_{\overline{b}},i}, S_s^{a_{\overline{b}},i}(c), b_s^{\overline{b},i},\ \forall s \in \tilde{\mathcal{S}}) =$$

$$= P(A_0 = V_{a_0}(i), A_1 = V_{a_1}(i) \mid S_s^{a_{\overline{b}},i}, I_s^{a_{\overline{b}},i}, S_s^{a_{\overline{b}},i}(c), b_s^{\overline{b},i}\ \forall s \in \tilde{\mathcal{S}}) =$$

$$= P(A_0 = V_{a_0}(i) \mid S_s^{a_{\overline{b}},i}, I_s^{a_{\overline{b}},i}, S_s^{a_{\overline{b}},i}(c), b_s^{\overline{b},i}\ \forall s \in \tilde{\mathcal{S}})$$

Since the random polynomials used to calculate each share belonging to $\mathcal{V}_{\overline{b}}, \mathcal{I}_{\overline{b}}, \mathcal{O}_{\overline{b}}$ are independently generated, the knowledge of $S_s^{a_{\overline{b}},i}$ gives no information about $A_{1-\overline{b}}$. Further, we note that $b_s^{\overline{b},i}$ are random bits independently chosen w.r.t. the secrets, and that the messages listed in $I_s^{a_{\overline{b}},i}$ are either shares of functions of the random numbers $r_s^{\overline{b},i}, r_s^{'\overline{b},i}, b_s^{\overline{b},i}$ (see Section 3.1), or of intermediate results for the collaborative computation of $S_s^{a_{\overline{b}},i}(c)$, in which each share is in turn divided in w shares according to the procedure described in [31]. Therefore, by exploiting the perfect secrecy property of SSS (which states that the knowledge of less than t shares does not leak any information about the secret), we can write:

$$P(A_0 = V_{a_0}(i) \mid S_s^{a_{\overline{b}},i}, S_s^{a_{\overline{b}},i}(c), I_s^{a_{\overline{b}},i}, b_s^{\overline{b},i}\ \forall s \in \tilde{\mathcal{S}}) = P(A_0 = V_{a_0}(i)) = P(\overline{b} = 0) = \frac{1}{2}$$

The extension to a set of $\tilde{T}$ measurements is straightforward. Note that for Appliances of the same type, the overall number of samples in a message is the same, thus the message length does not provide any additional information. □

Since in this paper we assume $t = w$, information leakages can occur only in case all the w Schedulers are compromised and the infrastructure is w-**blind**.

Theorem 3. *Under assumption that the Crowds message forwarding routine follows Algorithm 1, the privacy-friendly load scheduling architecture provides* ***sender anonymity****.*

Proof. Let $\tilde{\mathcal{G}}$ be the random variable indicating the set of Gateways from which the set of Schedulers $\mathcal{S}$ receive the w encrypted shares. Since each share is routed independently, then

$$P(\tilde{\mathcal{G}} = (x_1, x_2, \ldots, x_w)|G = g) = P(\tilde{G} = x|G = g)^w.$$

Using (1) we have,

$$P(\tilde{G} = x|G = g)^w = P(\tilde{G} = x)^w = P(\tilde{\mathcal{G}} = (x_1, x_2, \ldots, x_w))$$

Therefore, $\mathcal{D}$ obtains no additional information about the identifier of the local Gateway g, from which it follows that the `S-Anon` experiment has no advantage with respect to randomness. □

6. Benchmark Integer Linear Program

In order to evaluate the performance of our privacy-preserving scheduling approach, we propose as benchmark the following ILP model. It assumes to receive as input the time of arrival of each service request and the corresponding appliance load profile, within the time span considered for the allocation of the energy requests. Conversely, our scheduling infrastructure performs the allocation in real-time without having access to the individual energy consumption profile of the electrical appliances.

Sets

- $\mathcal{A}$: set of deferrable and interruptible Appliances
- $\mathcal{I} = 1, \cdots, \tilde{T}$: set of discretized time slots within the optimization time span
- $\mathcal{T} = \{1, \ldots, \max_a \tilde{D}_a\}$: set of discretized time slots within the runtime of the Appliances ($\max_{a \in \mathcal{A}} \tilde{D}_a < \tilde{T}$)

Parameters

- e_i: amount of RES-supplied energy at time $i \in \mathcal{I}$
- m_i: aggregated energy consumption profile of must-run Appliances
- t_a: time of arrival of the service request generated by Appliance $a \in \mathcal{A}$
- $\tilde{D}_a$: runtime of Appliance $a \in \mathcal{A}$
- k_{ai}: it is 1 if $i \geq t_a$, 0 otherwise
- c_{at}: load profile of Appliance $a \in \mathcal{A}$ at time $t \in \mathcal{T}$

Variables

- z_{ait}: binary variable, it is 1 if the t-th sample of appliance $a \in \mathcal{A}$ is scheduled at slot i, 0 otherwise

Objective function

$$\min \sum_{a \in \mathcal{A}} \left((\sum_{i \in \mathcal{I}} i z_{ai\tilde{D}_a}) - (t_a - 1) - \tilde{D}_a \right) \tag{8}$$

Constraints

$$\sum_{a \in \mathcal{A}, t \in \mathcal{T}:\ t \leq \tilde{D}_a} c_{at} z_{ait} \leq \begin{cases} e_i - m_i \text{ if } e_i > m_i \\ 0 \text{ otherwise} \end{cases} \quad \forall i \in \mathcal{I} \tag{9}$$

$$z_{ait} \leq k_{ai} \quad \forall a \in \mathcal{A}, i \in \mathcal{I}, t \in \mathcal{T}: t \leq \tilde{D}_a \tag{10}$$

$$\sum_{i \in \mathcal{I}} z_{ait} = 1 \quad \forall a \in \mathcal{A}, t \in \mathcal{T}: t \leq \tilde{D}_a \tag{11}$$

$$\sum_{j \in \mathcal{I}:\ j < i} z_{aj(t-1)} \geq z_{ait} \quad \forall a \in \mathcal{A}, i \in \mathcal{I}, t \in \mathcal{T}: t \leq \tilde{D}_a \tag{12}$$

The objective function (8) minimizes the sum of the delays experienced by the Appliances. Constraint (9) imposes that, in case the amount of energy required

by the must-run Appliances does not exceed the energy provided by RESes, the overall energy consumption due to both must-run and deferrable/interruptible Appliances does not go over the RES-provided energy level. Conversely, if the amount of energy required by must-run Appliances exceeds the energy production by RESes, no deferrable/interruptible Appliance is scheduled. Constraint (10) ensures that the Appliances' starting times are scheduled after the arrivals of the corresponding service requests, while Constraint (11) imposes that every (non-null) sample of each Appliance is scheduled exactly once. Finally, Constraint 12 ensures that the samples of each Appliance are scheduled sequentially (possibly interleaved by interruptions).

In order to adapt the above discussed model to deferrable non-interruptible Appliances, the following constraint has to be included in the formulation to impose that no intermediate service interruptions occur:

$$\sum_{i \in \mathcal{I}} iz_{ai\tilde{D}_a} - \sum_{i \in \mathcal{I}} iz_{ai1} + 1 = \tilde{D}_a \quad \forall a \in \mathcal{A} \tag{13}$$

7. Performance Evaluation

In this Section, we evaluate the performance of our proposed scheduling mechanism in terms of computational complexity, message number and length. Moreover, we compare the achieved average service delay to the optimal results obtained by means of the ILP formulation presented in Section 6. In our implementation, the hybrid cryptosystem used for the share encryption is the RSA-KEM Key Transport Algorithm [41], which uses the RSA public key cryptosystem modulo n, the KDF2 key derivation function (based on SHA-1) and the AES-Wrap-k key-wrapping scheme (where k is the AES key size) to communicate an ephemeral k-bit-long key used to encrypt the samples $V(i)$ by means of the standard AES scheme operating in Cipher Block Chaining mode (CBC).

7.1. Computational Complexity

We start discussing the asymptotic number of incoming/outgoing messages for each node and scheduling phase. As shown in Table 2, the number of mes-

sages exchanged by the Gateways exhibits a linear dependence on w, while for the Schedulers it depends linearly on $\tilde{T}$ and superlinearly on w (the logarithmic factor is due to the collaborative comparison procedure discussed in [33]). However, since the total number of shares w is expected to be limited, the time horizon $\tilde{T}$ is the only tunable parameters significantly influencing the number of messages.

Table 3 reports the type and number of operations performed by each node for the scheduling of a single service request. The computational cost of each operation is detailed in Table 4 based on [31, 33]. Assuming a few Schedulers (e.g. less than 5) and the usage of optimized hardware for fast modular multiplication[3], at the Gateways the most demanding operation is the RSA-AES hybrid encryption, with timings in the order of milliseconds for must-run appliances, of hundreds of milliseconds for deferrable non-interruptible appliances, and of tens of seconds for deferrable interruptible appliances (results obtained based on the data provided in [42, 43]). At the Schedulers, the computational time of the share collaborative comparison performed in multiple rounds depending on w is comparable to the running time of the hybrid encryption/decryption.

Note that Table 3 does not report the operations performed by the Crowds forwarding routine: for a detailed a discussion on the impact of $|\mathcal{G}|$ and p on the message latency, expected path length and number of appearances of a given Gateway on all paths, the reader is referred to [35].

Finally, it is worth discussing the message length. Let $L[x]$ indicate the length of message x in bits, and let $Pad[x]$ indicate the length in bits of the message of size x after PKCS1.5 cryptographic padding and concatenation to a 128-bit-long initialization vector, as required by the AES specifications. Each service request generated/forwarded by the Gateways and received by a Scheduler is an RSA-KEM encrypted message M_s of $L[n]+k+64+Pad[k+L[AT_a]+L[r_a]+\tilde{T}L[q]]$ bits in case of must-run Appliance (where $k+64$ bits is the output

[3]We estimated timings of tens of nanoseconds per single 64-bit modular Montgomery multiplication on a 2.5 GHz Intel Core i5 processor

Table 1: Message length for a service request generated at time slot $\tau = 1$.

Appliance Type	M_s	O_s	F_s
Must-run	19.90 kbit	18.85 kbit	19.78 kbit
Deferr.	5.31 Mbit	5.39 Mbit	19.78 kbit
Deferr. Interr.	1.53 Gbit	1.55 Gbit	19.78 kbit

length of the AES-Wrap-k key-wrapping scheme), of $L[n] + k + 64 + Pad[k + L[AT_a] + L[r_a] + (\tilde{T} - \tau)\tilde{T}L[q]]$ bits in case of deferrable non-interruptible Appliance, and of $L[n] + k + 64 + Pad[k + L[AT_a] + L[r_a] + (\tilde{T} - \tau)^2\tilde{T}L[q]]$ bits in case of deferrable and interruptible Appliance. During the share comparison procedure, each share received by a Scheduler is in turn divided in w shares, which are redistributed among the Schedulers. In case of deferrable non-interruptible (resp. deferrable and interruptible) Appliances, to perform $\tilde{T}(\tilde{T} - \tau)$ (resp. $\tilde{T}(\tilde{T} - \tau)^2$) comparisons per round each Scheduler sends/receives $w - 1$ messages per round of $\tilde{T}(\tilde{T} - \tau)L[q]$ (resp. $\tilde{T}(\tilde{T} - \tau)^2L[q]$) bits each (see [33] for further details). The comparison output O_s broadcasted by each Scheduler to every Gateway consists of w messages of $L[r_a] + Pad[(\tilde{T} - \tau)\tilde{T}(L[q] + 1)]$ (resp.$L[r_a] + Pad[(\tilde{T} - \tau)^2\tilde{T}(L[q] + 1)$). Finally, the Gateway sends to each Scheduler the selected consumption curve through the message F_s of length $L[n] + k + 64 + Pad[\tilde{T}L[q]]$.

Possible choices of the system parameters are: 1024-bit-long modulo n for the RSA cryptosystem, key length k =128 bits for the AES cryptosystem and 64-bit-long modulo q for the SSS scheme. The appliance tag AT_a and the random number r_a can be assumed to have length of 2 bits and 32 bits respectively. The assumed scheduling horizon can be a 24 hours period, divided in time slot duration is 5 minutes, which corresponds to $\tilde{T}$ =288 samples. The message lengths obtained with these parameters are summarized in Table 1 for the worst case of $\tau = 1$. Since in case of deferrable and interruptible Appliances the message size of M_s and O_s is in the order of Gb, a trade-off between the duration of the scheduling horizon and of the single time slots must be found (e.g. 12 hours with slots of 15 minutes would result in lengths of tens of Mb).

Table 2: Asymptotic number of incoming/outgoing messages per node for the scheduling of a single service request

Phase	Input	Output
Gateway		
Send request	$O(\frac{wp}{\lvert\mathcal{G}\rvert(1-p)})$	$O(\frac{wp}{\lvert\mathcal{G}\rvert(1-p)})$
Comparison computation (deferr./interr)	$O(w)$	-
Update consumption curve (deferr./interr)	$O(\frac{wp}{\lvert\mathcal{G}\rvert(1-p)})$	$O(\frac{wp}{\lvert\mathcal{G}\rvert(1-p)})$
Scheduler		
Send request	$O(1)$	-
Comparison computation (deferr.)	$O(w^2\lceil\log_2(w)\rceil\tilde{T}(\tilde{T}-\tau))$	$O(w^2\lceil\log_2(w)\rceil\tilde{T}(\tilde{T}-\tau))$
Comparison computation (interr.)	$O(w^2\lceil\log_2(w)\rceil\tilde{T}(\tilde{T}-\tau)^2)$	$O(w^2\lceil\log_2(w)\rceil\tilde{T}(\tilde{T}-\tau)^2)$
Update consumption curve (deferr./interr)	$O(1)$	-

Table 3: Computational load at each node for the scheduling of a single service request

Gateway	must-run: $\tilde{T}C_s(q) + wC_e(n, Pad[L[q]\tilde{T}])$
	def.: $\tilde{T}(\tilde{T}-\tau)(C_s(q)+C_l(q)) + wC_e(n, Pad[k + L[AT_a] + L[r_a] + L[q]\tilde{T}(\tilde{T}-\tau)]) + wC_d(n, Pad[(L[q]+1)\tilde{T}(\tilde{T}-\tau)]) + C_y(L[\tilde{T}]) + wC_e(n, Pad[L[q]\tilde{T}])$
	def. int.: $\tilde{T}(\tilde{T}-\tau)^2(C_s(q) + C_l(q)) + wC_e(n, Pad[k + L[AT_a] + L[r_a] + L[q]\tilde{T}(\tilde{T}-\tau)^2]) + C_d(n, Pad[(L[q]+1)\tilde{T}(\tilde{T}-\tau)^2]) + (\tilde{T}-\tau)C_y(L[\tilde{T}]) + wC_e(n, Pad[L[q]\tilde{T}(\tilde{T}-\tau)])$
Scheduler	must-run: $C_d(n, Pad[k + L[AT_a] + L[r_a] + L[q]\tilde{T}]) + \tilde{T}C_a(q)$
	def.: $C_d(n, Pad[k + L[AT_a] + L[r_a] + L[q]\tilde{T}(\tilde{T}-\tau)]) + \tilde{T}(\tilde{T}-\tau)(C_a(q) + C_c(q)) + \tilde{T}C_a(q) + C_e(n, Pad[(L[q]+1)\tilde{T}(\tilde{T}-\tau)]) + C_d(n, Pad[L[q]\tilde{T}])$
	def. int.: $C_d(n, Pad[k + L[AT_a] + L[r_a] + L[q]\tilde{T}(\tilde{T}-\tau)^2]) + \tilde{T}(\tilde{T}-\tau)^2(C_a(q) + C_c(q)) + \tilde{T}(\tilde{T}-\tau)C_a(q) + C_e(n, Pad[(L[q]+1)\tilde{T}(\tilde{T}-\tau)^2]) + C_d(n, Pad[L[q]\tilde{T}(\tilde{T}-\tau)])$

see Table 4 for the cost details.

Table 4: Detail of operation costs

Notation	**Description**	**Computational Cost**
$C_s(x)$	cost of the generation of w shares modulo x	$w(w-1)$ additions modulo x + $w(w-1)$ multiplications modulo x + $(w-1)$ random number generations modulo x
$C_a(x)$	cost of a share addition modulo x	1 addition modulo x
$C_l(x)$	cost of a share Lagrange interpolation modulo x	$O(w^2)$ multiplications modulo x
$C_m(x)$	cost of a share collaborative multiplication modulo x	2 multiplications modulo x $+C_s(x)$ + $C_a(x)$, performed in 2 rounds
$C_y(x)$	cost of finding the lowest feasible scheduling delay	$\tilde{T}(\tilde{T}-\tau)$ XOR operations over x bits + $\tilde{T}(\tilde{T}-\tau)$ AND operations over 1 bit + $(\tilde{T}-\tau)$ comparisons over x bits
$C_c(x)$	cost of a collaborative comparison modulo x	2 random number generation modulo x + 1 random number generation modulo 2 + 2 exponentiations modulo x + 2 multiplications modulo x + $2C_s(x)+(w+1)C_a(x)+O(w)C_m(x)$, performed in $\lceil \log_2 w \rceil$ rounds
$C_e(x,l)$	cost of an RSA-KEM encryption with RSA modulo x and AES encryption of a message of l bits	1 random number generation modulo x + 1 exponentiation modulo x 1 KDF2 key derivation and AES-Wrap-128 key wrapping + l AES encryptions
$C_d(x,l)$	cost of an RSA-KEM decryption with RSA modulo x and AES decryption of a message of l bits	1 exponentiation modulo x + 1 KDF2 key derivation and AES-Wrap-128 key unwrapping + l AES decryptions

7.2. Numerical Assessment

To compare the service delay introduced by our first-fit scheduling approach to the minimum delay obtainable through an optimization procedure, we extracted several load profiles of dishwashers (peak consumption of 1500 W), washing machines (peak consumption of 750 W), and dryers (peak consumption 6000 W) from the SMART* dataset [44] and resampled them with a rate of one sample every 5 minutes. As renewable energy supplying profile, we considered a windfarm with peak production of 50 MW: the normalized hourly production (available at [45]) has been linearly interpolated to obtain a 5 minutes sampling period. We considered a scenario with 20 households equipped with one dishwasher, one washing machine and one dryer each, for a total amount of 60 appliances. Each of them generates a service request with uniform distribution within a period of 24 hours, and 365 instances, corresponding to 1 year of wind energy production data. Each household is also equipped with a set of must-run appliances including lights, oven, fridge, and heater (see [44] for the comprehensive list), with a peak overall consumption of 5000 W.

For each instance, both the scheduling approach proposed in Section 4 and the ILP formulation described in Section 6 have been applied, first under the assumption that the 60 appliances are deferrable non-interruptible, then assuming them as deferrable and interruptible. Since the time horizon of each instance is one day, in case the scheduling delay of an Appliance exceeds 24 hours, the scheduling is considered to be infeasible. Table 5 reports the respective probabilities of finding a feasible solution to the scheduling problem. For non-interruptible Appliances, in approximately 15.1% of the considered instances, both approaches do not provide a feasible result: this happens when the overall daily energy production is not sufficient to satisfy all the service requests. Therefore, in those cases, the Appliances must be served using non-renewable energy sources, which are assumed to be unlimited and thus do not introduce any scheduling delay[4]. Such percentage reduces to 13.2% in case of interruptible

[4]We do not investigate a mixed RES/non-RES approach, which would introduce unfairness

appliances. In a borderline scenario, where the amount of wind energy is only slightly greater than the total energy demand, it may happen that our proposed scheduling approach fails in providing a feasible schedule, while the ILP formulation succeeds. However, we incurred in such condition only for the 0.8% of the considered instances in case of non-interruptible Appliances and for 1.9% in case of interruptible Appliances. Finally, in around 84% of cases, both approaches provide feasible solutions to the scheduling problem either for interruptible and non-interruptible Appliances: the average delay between service request and starting time experienced by a single appliance is in the range of 37-42 minutes, with an average increase of 1.8% of the suboptimal scheduling (maximum gap of 32.7% for interruptible Appliances and of 40.1% for non-interruptible Appliances) of our proposed infrastructure with respect to the optimal solutions obtained through the ILP model. With both approaches, in case of interruptible Appliances the scheduling delay is slightly reduced (2 mins per Appliance on average) with respect to the non-interruptible case. Therefore, our scheduling mechanisms protects users' privacy without significantly affecting the service delays experienced by the Appliances.

8. Conclusions

This paper proposes a privacy-preserving framework for the scheduling of power consumption requests generated by electrical Appliances in a Smart Grid scenario. To the best of our knowledge, this is the first attempt to address the problem of securely handling user data to provide a load scheduling service. The energy consumption requests generated by the smart Appliances located in the users' households within a neighborhood are anonymously conveyed to a set of Schedulers by means of a Crowds-based routing protocol. The Schedulers collaboratively define the schedule of the requests using a Multiparty Computation

in the service policy, since the fraction of Appliances powered with RES energy would possibly experience a scheduling delay, while the ones powered with traditional energy sources would be served immediately.

Table 5: Comparison of feasibility and average scheduling delay

Feasibility		occurrence [%]	Average Delay [min]		Gap [%]
Non-interruptible Appliances					
Secure	Optimal		Secure	Optimal	
✓	✓	84.1	41.7	39.8	1.9
✗	✓	0.8	-	147.4	-
✗	✗	15.1	-	-	-
Interruptible Appliances					
✓	✓	84.9	39.8	37.8	1.7
✗	✓	1.9	-	259.1	-
✗	✗	13.2	-	-	-

mechanism based on Shamir Secret Sharing scheme. We evaluate the security guarantees provided by our proposed infrastructure assuming an honest-but-curious attacker model and show through numerical results that the additional delay is modest with respect to the optimal solutions obtained by means of an Integer Linear Programming formulation.

References

[1] M. Alizadeh, X. Li, Z. W. et al., Demand-side management in the smart grid: Information processing for the power switch, Signal Processing Magazine, IEEE 29 (5) (2012) 55–67. doi:10.1109/MSP.2012.2192951.

[2] S. Bahramirad, H. Daneshi, Optimal sizing of smart grid storage management system in a microgrid, in: Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES, 2012, pp. 1–7. doi:10.1109/ISGT.2012.6175774.

[3] Y. Jia-hai, Customer response under time-of-use electricity pricing policy based on multi-agent system simulation, in: Power Systems Conference and Exposition, 2006 IEEE PES, 2006, pp. 814–818. doi:10.1109/PSCE.2006.296420.

[4] G. Hart, Nonintrusive appliance load monitoring, Proceedings of the IEEE 80 (12) (1992) 1870 –1891.

[5] C. Laughman, K. Lee, R. e. a. Cox, Power signature analysis, Power and Energy Magazine, IEEE 1 (2) (2003) 56 – 63.

[6] Committee on Homeland Security, Promoting and enhancing cybersecurity and information sharing effectiveness (PRECISE) act of 2011, Bill H.R. 3674 (Dec. 2011).
URL http://www.gpo.gov/fdsys/pkg/BILLS-112hr3674ih/pdf/BILLS-112hr3674ih.pdf

[7] A. Mohsenian-Rad, A. Leon-Garcia, Optimal residential load control with price prediction in real-time electricity pricing environments, Smart Grid, IEEE Transactions on 1 (2) (2010) 120–133. doi:10.1109/TSG.2010.2055903.

[8] P. Samadi, H. Mohsenian-Rad, V. Wong, R. Schober, Tackling the load uncertainty challenges for energy consumption scheduling in smart grid, Smart Grid, IEEE Transactions on PP (99) (2013) 1–10. doi:10.1109/TSG.2012.2234769.

[9] M. Alizadeh, A. Scaglione, R. Thomas, From packet to power switching: Digital direct load scheduling, Selected Areas in Communications, IEEE Journal on 30 (6) (2012) 1027–1036. doi:10.1109/JSAC.2012.120702.

[10] K. Jansen, G. Zhang, On rectangle packing: maximizing benefits, in: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '04, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2004, pp. 204–213.

[11] B. Baker, J. Schwarz, Shelf algorithms for two-dimensional packing problems, SIAM Journal on Computing 12 (3) (1983) 508–525.

[12] M. M. Karbasioun, G. Shaikhet, E. Kranakis, I. Lambadaris, Power strip packing of malleable demands in smart grid, CoRR abs/1302.3889.

[13] T.-H. Chang, M. Alizadeh, A. Scaglione, Real-time power balancing via decentralized coordinated home energy scheduling, Smart Grid, IEEE Transactions on 4 (3) (2013) 1490–1504. doi:10.1109/TSG.2013.2250532.

[14] P. Chavali, P. Yang, A. Nehorai, A distributed algorithm of appliance scheduling for home energy management system, Smart Grid, IEEE Transactions on 5 (1) (2014) 282–290.

[15] C. Chen, K. Nagananda, G. Xiong, S. Kishore, L. Snyder, A communication-based appliance scheduling scheme for consumer-premise energy management systems, Smart Grid, IEEE Transactions on 4 (1) (2013) 56–65.

[16] A.-H. Mohsenian-Rad, V. Wong, J. Jatskevich, R. Schober, A. Leon-Garcia, Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid, Smart Grid, IEEE Transactions on 1 (3) (2010) 320–331.

[17] Z. Wang, G. Zheng, Residential appliances identification and monitoring by a nonintrusive method, Smart Grid, IEEE Transactions on 3 (1) (2012) 80–92.

[18] M. Jawurek, M. Johns, K. Rieck, Smart metering de-pseudonymization, in: Proceedings of the 27th Annual Computer Security Applications Conference, ACSAC '11, ACM, New York, NY, USA, 2011, pp. 227–236. doi:10.1145/2076732.2076764.
URL http://doi.acm.org/10.1145/2076732.2076764

[19] D. Engel, Wavelet-based load profile representation for smart meter privacy, in: Innovative Smart Grid Technologies (ISGT), 2013 IEEE PES, 2013, pp. 1–6. doi:10.1109/ISGT.2013.6497835.

[20] E. Shi, T. Chan, Rieffel, R. Chow, D. Song, Privacy-preserving aggregation of time-series data, in: NDSS Symposium, 2011.

[21] C. Rottondi, M. Savi, D. Polenghi, G. verticale, C. Krauß, A decisional attack to privacy-friendly data aggregation in smart grids, submitted to *IEEE GLOBECOM* 2013.

[22] S. Rajagopalan, L. Sankar, S. Mohajer, H. Poor, Smart meter privacy: A utility-privacy framework, in: Smart Grid Communications, 2011 IEEE International Conference on, 2011, pp. 190 –195.

[23] C. Rottondi, G.Mauri, G. Verticale, A data pseudonymization protocol for smart grids, in: IEEE GreenCom Online Conference on Green Communications, 2012.

[24] C. Efthymiou, G. Kalogridis, Smart grid privacy via anonymization of smart metering data, in: Smart Grid Communications, 2010 First IEEE International Conference on, 2010, pp. 238 –243. doi:10.1109/SMARTGRID.2010.5622050.

[25] T. Dimitriou, G. Karame, Privacy-friendly tasking and trading of energy in smart grids, in: Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, ACM, New York, NY, USA, 2013, pp. 652–659. doi:10.1145/2480362.2480488.
URL http://doi.acm.org/10.1145/2480362.2480488

[26] A. Rial, G. Danezis, Privacy-preserving smart metering, in: Proceedings of the 10th annual ACM workshop on Privacy in the electronic society, WPES '11, ACM, New York, NY, USA, 2011, pp. 49–60. doi:http://doi.acm.org/10.1145/2046556.2046564.
URL http://doi.acm.org/10.1145/2046556.2046564

[27] K. Kursawe, G. Danezis, M. Kohlweiss, Privacy-friendly aggregation for the smart-grid, in: Privacy Enhancing Technologies, Vol. 6794, Springer Berlin / Heidelberg, 2011, pp. 175–191.

[28] C. Rottondi, G. Verticale, A. Capone, Privacy-preserving smart metering

with multiple data consumers, Computer Networks 57 (7) (2013) 1699 – 1713.

[29] M. Burkhart, M. Strasser, D. Many, X. Dimitropoulos, SEPIA: Privacy-preserving aggregation of multi-domain network events and statistics, in: USENIX SECURITY SYMPOSIUM, USENIX, 2010.

[30] A. Shamir, How to share a secret, Commun. ACM 22 (1979) 612–613.

[31] D. Bogdanov, Foundations and properties of shamir's secret sharing scheme, research Seminar in Cryptography (2007).
URL http://research.cyber.ee/~peeter/teaching/seminar07k/bogdanov.pdf

[32] T. Nishide, K. Ohta, Multiparty computation for interval, equality, and comparison without bit-decomposition protocol, in: Proc. of the 10th international conference on Practice and theory in public-key cryptography, PKC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 343–360.

[33] F. Kerschbaum, D. Biswas, S. de Hoogh, Performance comparison of secure comparison protocols, in: Database and Expert Systems Application, 2009. 20th International Workshop on, 2009, pp. 133–136. doi:10.1109/DEXA.2009.37.

[34] F. Kerschbaum, O. Terzidis, Filtering for private collaborative benchmarking, in: G. Mller (Ed.), Emerging Trends in Information and Communication Security, Vol. 3995 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2006, pp. 409–422. doi:10.1007/11766155_29.
URL http://dx.doi.org/10.1007/11766155_29

[35] M. K. Reiter, A. D. Rubin, Anonymous web transactions with crowds, Commun. ACM 42 (2) (1999) 32–48.

[36] Federal Office for Information Security, Protection profile for the gateway of a smart metering system (2011).

URL https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/SmartMeter/PP-SmartMeter.pdf

[37] J. Katz, Y. Lindell, Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series), Chapman & Hall/CRC, 2007.

[38] A. Pfitzmann, M. Hansen, Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management – a consolidated proposal for terminology, v0.31 (Feb. 2008).

[39] D. Stinson, Cryptography Theory and Practice, Second Edition, CRC Press, 2005.

[40] C. Rottondi, G. Mauri, G. Verticale, A protocol for metering data pseudonymization in smart grids, Transactions on Emerging Telecommunications Technologies (2013) n/a–n/adoi:10.1002/ett.2760.
URL http://home.dei.polimi.it/vertical/papers/2013_ett.pdf

[41] Internet Engineering Task Force (IETF), Use of the RSA-KEM key transport algorithm in the cryptographic message syntax (CMS), request for comments: 5990 (Sept. 2010).

[42] Hardware AES showdown - VIA padlock vs intel AES-NI vs AMD hexacore.
URL http://grantmcwilliams.com/tech/technology/item/532-hardware-aes-showdown-via-padlock-vs-intel-aes-ni-vs-amd-hexacore

[43] S. Gueron, V. Krasnov, Software implementation of modular exponentiation, using advanced vector instructions architectures, in: F. zbudak, F. Rodrguez-Henrquez (Eds.), Arithmetic of Finite Fields, Vol. 7369 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 119–135. doi:10.1007/978-3-642-31662-3_9.
URL http://dx.doi.org/10.1007/978-3-642-31662-3_9

[44] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, J. Albrecht, Smart*: An open data set and tools for enabling research in sustainable homes, in: The 1st KDD Workshop on Data Mining Applications in Sustainability (SustKDD), 2011.

[45] Global energy forecasting competition 2012 - wind forecasting.
URL http://www.kaggle.com/c/GEF2012-wind-forecasting/data