Privacy-friendly Aggregation for the Smart-grid

Klaus Kursawe[1], George Danezis[2], and Markulf Kohlweiss[2]

[1]Radboud Universiteit Nijmegen,
kursawe@cs.ru.nl

[2]Microsoft Research, Cambridge, U.K.
{gdane, markulf}@microsoft.com

Abstract. The widespread deployment of smart meters for electricity gas and water consumption to modernise the electricity systems, has been associated with privacy concerns. In this paper, we present protocols that can be used to privately compute aggregate meter measurements, allowing for fraud and leakage detection as well as further statistical processing of meter measurements, without revealing any additional information about the individual meter readings.

1 Introduction.

Smart-grid deployments are actively promoted by many governments, including the United States as well as the European Union. Yet, current smart metering technologies rely on centralizing personal consumption information, leading to privacy concerns. We address the problem of security aggregating meter readings without the provider learning any information besides the aggregate, or to compare an aggregate with a known value to detect fraud or leakage (the latter is more relevant for water and gas metering).

Fraud detection is a major issue for electricity metering, and will be one significant use-case in the upcoming smart grid. A recent FBI report[1] states that spot checks in one state have shown 10% of all smart meters to have been tampered with. Aggregates of consumption across different populations are also used for forecasting, tuning production to demand, settling the cost of production across electricity suppliers, and getting a clear picture on the supply of consumer generated energy, e.g., through solar panels. Aggregation protocols will also be used to detect leakages in other utilities, e.g., water (which is a big issue in desert countries) and gas (where a leakage poses a safety problem).

Privacy in Smart Metering. The area of smart metering for electricity, but also other commodities such as gas and water is currently experiencing a huge push; for example, the European commission has formulated the goal to provide 80% of all households with smart electricity meters by the year 2020 [1], and the US government has dedicated a significant part of the stimulus package towards

[1] Obtained through personal communication.

a smart grid implementation. Simultaneously, privacy issues are mounting – in 2009, the Dutch Senate stopped a law aimed to make the usage of smart meters compulsory based on privacy and human rights issues [2]. On the US side, NIST has identified privacy as one of the main concerns in a smart grid implementation, and proposes using the "privacy by design" approach [3] to alleviate them. While it is not clear yet how much data can be derived from actual meter readings, the high frequency suggested (i.e., about 15 minute reading intervals), together with the difficulty to temporarily hide one's behaviour (as one can do, for example, by turning off a mobile phone), gives rise to serious privacy concerns. For water and gas leakage detection privacy preserving protocols are even more desirable since measurements need to be frequent to detect potentially dangerous leaks as soon as possible.

An important aspect in privacy preserving metering protocols is to take into account the rather limited resources on such meters, both in terms of bandwidth and in terms of computation. We therefore push as much workload as possible to the back-end, leaving the minimal work possible on the meter itself. In terms of communication, the messages sent out by the meters should increase only minimally. Furthermore, meters should ideally act independently, without requiring interaction with other meters wherever possible and minimal interaction when not.

For statistical analysis, out protocols support the division of meters into independent sets over which the aggregation is to be done. This allows for different use-cases that require only statistical accuracy to be combined without any additional effort on the meters. To validate the practicality of our protocols in a real setting, a proof-of-concept implementation is currently underway in collaboration with a meter manufacturer and a Dutch utility.

Related work. Privacy preserving metering aggregation and comparison has been introduced by Garcia and Jacobs [4]. Their protocol requires $O(n^2)$ bytes of interaction between the individual meters as well as relatively expensive cryptography on the meters (Paillier ecnryption). Fu. et all [5], highlight the privacy related threats of smart metering and propose an architecture for secure measurements, that rely on trusted components outside of the meter. Rial and Danezis [6] propose a protocol using commitments and zero knowledge proofs to privately derive and prove the correctness of bills, but not for aggregation across meters. The latter techniques have also been extended to protocols that provide differential privacy guarantees [7].

2 Basic Protocols

The protocols we propose follow the principle of [8] by relying on masking the meter consumptions $c_{i,j}$ output by meter j for a reading i, in such a way that an adversary cannot recover individual readings. Yet, the sum of the masking values across meters sums to a known value (for simplicity we set it to be zero here; however, in a practical setting, a non-zero value may allow for aggregating

over several different sets of meters and easier group management). As a result summing the masked readings uncovers their sum or a one-way function of their sum. To prevent linking masked values, the masks are recomputed for every measurement either by a symmetric protocol with communication between the meters, or by an asymmetric one that does not require such. We refer to the combination of a meter and a user as a metered home, or home in short. We consider two types of protocols:

In the first, which we refer to as *aggregation protocols*, metered homes use masking values $x_{i,j}$ to output blinded values $x_{i,j} + c_{i,j}$. After the masking values have canceled each other out, the result of the protocol is $\sum c_{i,j}$.

In the second type of protocols, homes output $g_i^{x_j + c_{i,j}}$ and the result of the protocol is $g_i^{\sum c_{i,j}}$. We call the latter protocols *comparison protocols*, because they require that the aggregator already knows the (approximate) sum of the values she is aggregating (through a feeder meter), and needs to determine whether her sum is sufficiently close to the aggregate obtained from home meters. However, as shown in Section 4.6, the comparison protocol can easily be turned into a full aggregation protocol with low overhead. In both cases we assume that the output of homes that is aggregated preserves the authenticity of $c_{i,j}$.[2]

Comparison protocols offer advantages for cryptographic protocol design, as protocol values can be exponents in cryptographic groups for which the computation of discrete logarithms are in general hard. One advantage that can be garnered from this is that in contrast to aggregation protocols, no fresh $x_{i,j}$ are needed. As part of our security analysis, we show in Appendix A, that for random x_j and g_i, ${g_i}^{x_j}$ are indistinguishable from $g_i^{x_{i,j}}$, where the $x_{i,j}$ are chosen freshly for each g_i, under the Decisional Diffie-Hellman assumption.

The basic comparison protocol. Let $\mathbb{G}$ be a suitable Diffie-Hellman group, and $H : \{0,1\}^* \to \mathbb{G}$ a hash function mapping arbitrary strings onto elements of $\mathbb{G}$.[3] Let x_j be a pre-shared secret for home j such that $\sum_j x_j = 0$. We assume that each measurement round has a unique identifier i that is shared by all homes and the aggregator, e.g., a serial number or the time and date of the measurement.

For each reading $c_{i,j}$, the home computes a common group element $g_i = H(i)$. It then computes $g_{i,j} = {g_i}^{c_{i,j} + x_j}$. The value $g_{i,j}$ is then send to the aggregator. The aggregator collects all values of $g_{i,j}$, and computes $g_a = \prod_j g_{i,j}$.

By construction, we have $\prod_i g_{i,j} = \prod_i g_i^{c_{i,j}} \cdot \prod_i g_i^{x_i} = g^{\sum_i c_{i,j}}$, i.e., g_a is g_i to the power of the aggregated measurements. As the aggregator has it's own measurement c_a of the total consumption of the connected meters, it now needs to verify if g_a roughly equals g^{c_a}. This can be done by brute forcing values of $g^{c_a}, g^{c_a - 1}, g^{c_a + 1}, ...$ until either a match is found or a sufficiently large interval has been tested to raise an alarm.

---

[2] This can either be achieved by signing $x_{i,j} + c_{i,j}$ respectively $g_i^{x_j + c_{i,j}}$ with the meters secret key, or by using cryptographic verifiability as discussed in Section 4.1.

[3] For our security analysis we will make use of the random oracle model to guarantee the randomness of the g_i values [9].

3 Concrete Protocols

As we have seen, the general framework of our protocols requires a number of meters or users to have a secret value x_j per meter or $x_{i,j}$ per meter per round, such that they all add up to zero. Then the aggregation protocols can be used by each party publishing $x_{i,j} + c_{i,j}$, or the comparison protocol by publishing $g_i^{x_j+c_{i,j}}$. Concrete protocols provide different ways for a number of meters or users to derive the necessary $x_{i,j}$ or $g_i^{x_j}$.

We propose four such protocols each with different advantages: (1) a protocol that offers unconditional security based on secret sharing; (2,3) two protocols based on Diffie-Hellman key exchange that allow blinding to be verifiably done outside the meter; (4) finally a protocol based on computations on the meter, but with negligible communication overhead.

3.1 Interactive protocol.

Our first protocol uses simple additive secret sharing. For each round i of measurements, a subset of the homes is (deterministically) chosen as *leaders*[4]; all parties compute completely random secret shares, encrypt them, and send them to the leaders. The leaders then computes their final shares in a way that all shares together sum to zero. Shares at each home are added together with the meter reading to mask it; an aggregator can sum up all shares such that they cancel out and reveal the sum of all consumption across the homes.

More formally, we assume an aggregation set of n homes and one aggregator (substation). We call p the privacy parameter; this is the number of leaders in a run of the protocol. Note that for $p = n$ the interactive protocol has the same collusion security as [4]. At system setup, each home has its own private encryption key K_j, as well as the public encryption keys $PK_1, \ldots, PK_n$ for all other homes in the same aggregation set.

- To generate masking values, each home j first computes p random values $s_{j,1}, ..., s_{j,p}$. It then computes the leader identities $\ell_1, ..., \ell_p$ of the p leaders, and encrypts $s_{j,k}$ with PK_{ℓ_k}, $1 \leq k \leq p$. The set of p encrypted shares is sent to the aggregator that sends each leader its corresponding encrypted shares.
- Each leader ℓ_k collects $n-1$ shares $s_{j,k}$, $1 \leq j \leq n$, $j \neq \ell_k$, and computes its own share $s_{\ell_k,k}$ such that all shares together sum to the value 0 (modulo 2^{32}).
- Finally, all parties add all their shares $s_{j,1}, ..., s_{j,p}$ to get the main share s_j.

For the basic aggregation protocol, $x_{i,j} = s_j$. To update the masking values, the above steps are repeated with a different set of leaders for each reading i; the results for each meter is added to it's current share. To send a reading $c_{i,j}$, a

[4] Alternatively, leaders could be trusted third parties that do not contribute any consumption values themselves.

meter computes $b_{i,j} = c_{i,j} + s_{i,j} \bmod 2^{32}$. The aggregator collects all this data, and computes $\sum_i b_{i,j} = \sum_i c_{i,j}$.

The interactive protocol can also be used in combination with the basic comparison protocol by setting $x_j = s_j$, removing the need for updating shares.

3.2 Diffie-Hellman Key-Exchange Based Protocol.

Our second scheme is based on the standard Diffie-Hellman key exchange protocol, combined with a modified variant of the Dining Cryptographer's anonymity protocol [10, 11]. We assume that each meter j has a secret key X_j, and a corresponding public key Pub_j.

- For each round i, let $g_i = H(i)$ be a generator of a Diffie-Hellman group $\mathbb{G}$. The generator g_i is the same as for the basic comparison protocol.
- In the first phase of the protocol, each home computes a round specific public key $\mathrm{Pub}_{i,j} = g_i^{X_j}$, certifies it, and distributes it to all other members of the aggregation set.
- Homes receive and verify public keys $\mathrm{Pub}_{i,1}, \ldots, \mathrm{Pub}_{i,n}$.
- Each home can now compute the following value:

$$g_i^{x_j} = \prod_{k \neq j} \mathrm{Pub}_{i,k}^{(-1)^{k<j} X_j} ,$$

  where $k < j$ is an indicator variable taking value 1, if the name/index of meter k is lexicographically smaller than the name of meter j, and zero otherwise. As required the sum of all x_j is equal to 0:

$$\sum_j x_j = \sum_j \sum_{k \neq j} (-1)^{k<j} p_k \cdot p_j = 0 .$$

- Therefore each meter can compute $g_{i,j}$ as required by the comparison protocol as: $g_{i,j} = g_i^{c_{i,j}} \cdot g_i^{x_j} = g_i^{c_{i,j}+x_j}$.

Note that x_j cannot be known or recovered by any of the meters. This precludes the use of this protocol as an aggregation protocol, but is not an impediment to using it as a comparison protocol.

3.3 Diffie-Hellman and Bilinear-map Based Protocol.

The DH-based scheme can be extended to only require a fixed public key per meter. The construction is similarly to the modified Dining-Cryptographers protocols in [12]. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups in which the Decisional Bilinear Diffie-Hellman assumption [13] holds with a bi-linear map function $e(\mathbb{G}_1, \mathbb{G}_2) \to \mathbb{G}_T$. Each meter only has to produce once a fixed public key $\mathrm{Pub}_j = \hat{g}_0^{X_j}$ where $\hat{g}_0$ is a generator of $\mathbb{G}_1$. Let $H(\{0,1\}^*) \to \mathbb{G}_2$ be a hash function mapping arbitrary strings onto elements of $\mathbb{G}_2$.

– In round i, compute $\hat{g}_i = H(i)$ and $g_i = e(\hat{g}_0, \hat{g}_i)$. Homes can now compute $g_i^{x_j}$ as:

$$g_i^{x_j} = \left(\prod_{k \neq j} e(\mathrm{Pub}_k, \hat{g}_i)^{(-1)^{k<j}} \right)^{X_j} ,$$

where $k < j$ is an indicator variable taking value 1 or 0 depending on the result of the comparison. As required the sum of all x_j is 0:

$$\sum_j x_j = \sum_j \sum_{k \neq j} (-1)^{k<j} p_k \cdot p_j = 0 .$$

– Therefore each meter can compute $g_{i,j}$ as required by the comparison protocol as: $g_{i,j} = g_i^{c_{i,j}} \cdot g_i^{x_j} = g_i^{c_{i,j}+x_j}$.

Note that as in the pure Diffie-Hellman protocol x_j cannot be known or recovered by any of the meters. This is not an impediment to using it as a comparison protocol. As noted by [12], the map e can be instantiated with the Weil pairing over a suitable elliptic curve.

3.4 Low-overhead protocol.

As for the Bilinear map based scheme, we assume that all meters have a fixed public key $\mathrm{Pub}_j = g^{X_j}$ where g is a fixed globally known generator of a group in which the Computational Diffie-Hellman assumption holds.

– Each meter is initialised with the public keys of all other meters, and computes a set of shared keys, as: $K_{j,k} = H(\mathrm{Pub}_k^{X_j})$ Once the set of shared keys have been computed the original public keys of the other meters can be discarded.
– For each round i of masking value generation each meter j outputs:

$$x_{i,j} = \sum_{k \neq j} (-1)^{k<j} H(K_{j,k} \| i) .$$

For the basic aggregation protocol, only 32 bits of $x_{i,j}$ are needed, and $b_{i,j} = c_{i,j} + x_{i,j} \bmod 2^{32}$. The values $b_{i,j}$ are short 4 byte unsigned integers, and the aggregator can compute the sum simply by adding all the outputs together $\sum_j c_{i,j} = \sum_j b_{i,j} \bmod 2^{32}$.

The low-overhead protocol can also be used in combination with the basic comparison protocol by setting $x_j = x_{i',j}$ for a fixed i'. This removes the need for creating additional masking values. To allow for cryptographic verification of correct computation of $g_{i,j} = {g_i}^{c_{i,j}+x_j}$, the meter can output a commitment $g^{x_j} h^{open_{x_j}}$ together with a signature σ_{x_j} on this commitment under the meter's secret key.

4 Comparison between concrete protocols.

We proposed four concrete protocol variants to achieve private aggregation or comparison. In this section we compare them with regards to *cryptographic verifiability*, *cost & performance*, *availability*, *forward secrecy*, *group management*, *interoperability with other protocols* and finally their applicability to *further applications*.

4.1 Cryptographic Verifiability

The metering setting presented so far includes meters and an aggregator jointly computing the sum of consumption or comparing it to a known value. In practice meters are resource constraint devices in terms of memory, bandwidth, latency and storage, and to a lesser extent computation. Furthermore the architecture of smart-meters separates the certified metrological core, from other functions such as any user interface or communications logic, further constraining resources available for privacy protocols. For these reasons it might be beneficial to perform the bulk of any computations necessary for the aggregation protocol outside the meter or at least outside the certified metrological unit. Yet, despite off-loading those computations on untrusted hardware, under the control of the customer, we would like to ensure the correctness of the protocols – namely that the sum extracted through the aggregation protocol is indeed the sum of all readings from the meters.

Existing privacy-reserving billing protocols [6] have proposed a simple modification to meters that enables further privacy preserving computations: meters output commitments to their readings (such as Petersen commitments [14] of the form $C_{c_{i,j}} = g^{c_{i,j}} h^{\text{open}_{i,j}}$) and a signature over them. The customer associated with meter can open those commitments but can also use them as input to certify further computations. Let us evaluate how our proposed protocols are amenable to such certification.

In the context of verification we consider a meter, a customer, and an aggregator. The meter outputs signed commitments to its readings, as well as the raw readings to the customer. The customer performs the necessary steps of the aggregation or comparison protocol, but also outputs a universally verifiable cryptographic proof that protocol messages are correct. The aggregator receives the inputs of all customers, and can use the certified readings as well as the proof of all messages to ensure no customer has deviated from the valid protocol.

We use several existing results to prove statements about discrete logarithms, such as, proofs of knowledge of a discrete logarithm [15] and proofs of knowledge of the equality of elements in different representations [16]. These results are often given in the form of Σ-protocols but with the help of hash functions they can be turned into non-interactive zero-knowledge arguments in the random oracle model [17]. When referring to the proofs above, we follow the notation introduced by Camenisch and Stadler [18].

The *interactive protocol* can be verified by using a simple version of a verifiable secret sharing scheme [14] to certify that all protocol messages are well

formed. For every round of aggregation i each customer outputs a commitment $C_{x_{i,j}}$ to a random value $x_{i,j}$, as well as commitments $C_{s_{j,k}}$ to the shares $s_{j,k}$. Then it provides a proof in zero-knowledge that the sum of the shares is equal to the committed random value, and that the output value $c_{i,j} + x_{i,j}$ is indeed the sum of the random value and the genuine meter reading. Each leader further proves that their random share $s_{i,k}$ added to all the shares they received sums to the value zero. The proofs only involve statements about revelation of commitments and sums of commitments and are extremely efficient if a commitment scheme with an additive homomorphism is used, such as Petersen commitments.

The *DH based protocol* is also amenable to cryptographic verification. The customer can produce the value $g_{i,j}$ along with a certificate to prove it is correctly formed given their public key $\mathrm{Pub}_j = g^{X_j}$ and the commitment to the meter reading $C_{c_{i,j}}$. First, the customer needs to create a new public key using the generator g_i associated with the reading time i, and prove that it has the same secret key X_j. This public key $\mathrm{Pub}_{i,j}$ is published for all to retrieve.

Then using the public keys $\mathrm{Pub}_{i,k}$ of all other customers k, it needs to prove that the value $g_{i,j}$ is well formed given its own secret key. This involves a standard zero-knowledge proof that:

$$\mathrm{NIZK}(X_j, c_{i,j}, \mathrm{open}_{i,j})\{\mathrm{Pub}_j = g^{X_j} \wedge \mathrm{Pub}_{i,j} = g_i^{X_j} \wedge C_{c_{i,j}} = g^{c_{i,j}} h^{\mathrm{open}_{i,j}} \wedge g_{i,j} = g_i^{c_{i,j}} \cdot \left(\prod_{k \neq j} \mathrm{Pub}_{i,k}^{(-1)^{i<j}} \right)^{X_j} \} .$$

The *bilinear map based protocol* can also be verified cryptographically. Each meter has to prove that the value $g_{i,j}$ is formed correctly. This can be done efficiently with a proof that:

$$\mathrm{NIZK}(X_j, c_{i,j}, \mathrm{open}_{i,j})\{\mathrm{Pub}_j = \hat{g}_0^{X_j} \wedge C_{c_{i,j}} = g^{c_{i,j}} h^{\mathrm{open}_{i,j}} \wedge g_{i,j} = g_i^{c_{i,j}} \left(\prod_{k \neq j} e(\mathrm{Pub}_k, \hat{g}_i)^{(-1)^{k<j}} \right)^{X_j} \} .$$

This is similar to the proofs in [12], except that we do not have to worry about collisions in the Dining Cryptographers protocol. In fact, our protocol presupposes that every home contributes some value $g_i^{c_{i,j}}$ as a contribution to the sum $\sum_i c_{i,j}$.

Finally the *low-overhead protocol* is based on symmetric key primitives that do not exhibit the mathematical relations necessary for efficient zero-knowledge proofs. While it could in theory be cryptographically verified though decomposing it into a circuit, this would not be a practical protocol. Therefore this protocol has to be run within the trusted meter hardware.

When using the low-overhead protocol together with the basic comparison protocol some amount of cryptographic verifiability is possible. Cryptographic verifiability can, however, be guaranteed only for the correct construction of $g_{i,j}$

	Initialization	Communication	Computation
Interactive (agg)	$O(N^2) \cdot PK$	$O(N \cdot p) \cdot \mathbb{Z}_q$	$O(p) \cdot \mathsf{Enc}$
Interactive (comp)	$O(N^2) \cdot PK$	$O(N) \cdot \mathbb{G}$	$O(1) \cdot E$
	$+O(N \cdot p) \cdot \mathbb{Z}_q$		
DH	$O(N^2) \cdot \mathbb{G}$	$O(N^2) \cdot \mathbb{G}$	$O(N) \cdot M + O(1) \cdot E$
Pairing	$O(N^2) \cdot \mathbb{G}$	$O(N) \cdot \mathbb{G}$	$O(N) \cdot P + O(1) \cdot E$
Low-overhead (agg)	$O(N^2) \cdot \mathbb{G}$	$O(N) \cdot \mathbb{Z}_{2^{32}}$	$O(N) \cdot H$
GC [4]	$O(N^2) \cdot PK$	$O(N^2) \cdot \mathbb{Z}_{n^2}$	$O(N) \cdot \mathsf{Enc} + O(1) \cdot \mathsf{Dec}$

Table 1. Performance comparison: PK.. size of public keys, $|\mathbb{Z}_x|$, $\mathbb{G}$.. size of algebraic group, Enc, Dec, E, M, H.. cost of encryption, decryption, exponentiation, multiplication, or hash function evaluation respectively.

from the values committed in signed commitments C_{x_j} and $C_{c_{i,j}}$. This can be done efficiently with a proof that:

$$\begin{aligned} \mathrm{NIZK}(x_j, \mathrm{open}_{x_j}, c_{i,j}, \mathrm{open}_{i,j})\{C_{c_{x_j}} = g^{x_j} h^{\mathrm{open}_{x_j}} \\ \wedge C_{c_{i,j}} = g^{c_{i,j}} h^{\mathrm{open}_{i,j}} \wedge g_{i,j} = g_i^{x_j + c_{i,j}}\} \, . \end{aligned}$$

This might be useful for aggregating values that are not known to the meter (such a demographics, e.g. the number of people sharing a home). In such cases the meter can provide a signed commitment that is augmented by another certified item outside the meter.

4.2 Computation & Communication Overheads.

Whether the proposed protocols are executed by meters or by customers our protocols always impose some overhead over a privacy invasive solution.

The DH based protocol in its most secure form is the most expensive protocol, requiring $O(N^2)$ total messages to be exchanged as all participants need to have access to a new set of DH public keys $\mathrm{Pub}_{i,j}$ for the aggregation of each meter reading. A related version of the protocol could allow participants to only share keys with p other participants reducing the communication cost to $O(N \cdot p)$. The protocol requires $O(N)$ modular multiplications but only $O(1)$ exponentiations per participant.

The interactive protocol only requires $O(N \cdot p)$ messages to be sent from the normal participants to the leaders, and a further $O(p)$ messages from the leaders. The setup cost requires public key distribution which could cost from $O(N^2)$ messages to $O(N \cdot p)$ if leader are fixed. Computations are very fast as they only involve addition over large integers, but secrecy of shares forces each participant to perform $O(p)$ public key encryptions and each leader $O(N)$ decryptions. Its cryptographic proof can use homomorphisms involving multiplications and $O(1)$ exponentiations for each customer.

The pairing based scheme is the most economical in terms of communication overhead. The key distribution setup requires $O(N^2)$ messages for all homes to

be made aware of the long term public keys of all other meters. After that for each reading only $O(N)$ messages are required from the meters to the aggregator. Each participant needs to perform $O(N)$ pairing operations and $O(1)$ exponentiations.

The low-overhead protocol has to be run within the meter but is extremely compact and computationally efficient. Key distribution requires a one-off exchange of public keys which costs overall $O(N^2)$ messages and $O(N)$ exponentiations per participant. Subsequently, only $O(N)$ hash function applications are required, and only $O(N)$ small integer values are transmitted to the aggregator. This is the same communication cost as today's meters – giving the final protocol its name. We summarize the asymptotic performance of our protocols in Table 1 and compare it with [4]. We provide an experimental evaluation of this protocol in Section 5.

4.3 Availability, Privacy & Forward Secrecy

Considerations of whether to run the protocols in the meter or over customer hardware need to take into account the need for availability, or the principle "utility robustness" as it is known in the energy industry. The principle means that all parts necessary for the correct functioning of the energy supply system, including fraud detection, should be under the control of the energy industry. The key fear is that the energy supplier may not have the authority to replace a component when it fails, or is disabled. Therefore when the aggregation and comparision protocols are used for critical monitoring it is advisable to run them in the meters. When they are only used for non-critical tasks (such as tuning seasonal profiles of consumption) they can be off-loaded on customer machines and performed when the user is on-line.

Privacy is a key property of our protocols and it is maintained as long as all participants are honest-but-curious and do not collude. In case of passive collusion different protocols provide different guarantees. The DH based protocol, the bilinear maps based protocol, and the low-overhead protocol ensure that the anonymity set within which meter readings are aggregated includes all the non colluding meter readings. The interactive protocol has a similar property for any number of colluding nodes that does not include all leaders. If all leaders collude all privacy is lost.

Active attackers, that can break their meters, can disrupt the protocol so that the reported aggregate is different than the actual sum of consumptions. This is, however, at the heart of the fraud detection mechanism: the total may be different and thus has to be compared with the aggregator meter. Colluding attackers can also shift their reported consumption to appear as if some are consuming more or less subject to the sum being equal. While this attack does not change the total energy consumed it might still be beneficial for customers with variable tariffs. In case cryptographically verifiable protocols are used active adversaries should not be able to interfere with the integrity of the protocol messages unless they have compromised the physical meters, or have physically bypassed the meter – which is common.

Forward secrecy [19, 13, 20] is desirable to minimize the impact of a potentially leaked private key. The interactive and DH based protocols can be modified to provide some forward secrecy. The interactive protocol participants can use ephemeral keys to encrypt shares sent to the leaders, that are forgotten after a certain epoch. Similarly fresh DH keys can be used for each round of aggregation using the DH protocol, by signing them with the long term keys instead of proving they are the same. The overhead to modify the protocols in this manner is not high, since they already require $O(N^2)$ messages per round. On the other hand it is difficult to modify either the Bilinear map based protocol or the low-overhead protocol to provide forward secrecy while keeping their messages volumes at a similar level. Re-keying these protocols will require a fresh setup and $O(N^2)$ messages.

4.4 Key Establishment & Group Management

All proposed protocols require participants to be aware of the keys of meters, and other participants, including signature keys and encryption keys. In all cases we assume that meters contain a signature key to authenticate genuine messages. A private decryption key is used by some protocols to either communicate with leaders or build secure channels. These can be shared with the customers.

In case cryptographic certification is used to off-load computations a further secure channel is required between customers and meters to ensure only authorised customers can open the certified commitments to readings. In that case meters do not need to be aware of the keys of other parties, keeping them cheap.

Setup phases when keys are exchanged take from $O(l \cdot N)$ messages for the interactive protocol to $O(N^2)$ messages for the other protocols. For the bilinear maps based protocol and the low-overhead protocol this is a one-off cost, after which only $O(N)$ messages need to be exchanged.

In some cases keys will have to be rotated, either to ensure forward secrecy (as for example when the owner of a house changes) or to introduce or retire meters to groups. Adding, changing, or removing the key of a meter from a group only requires $O(N)$ messages, to notify all participants of the new certified key.

The security of the proposed schemes depends on the compositions of the meter groups. As we have already discussed a single honest participant within a group that is totally controlled by the adversary cannot expect any privacy. For this work we assume that the energy industry is in charge of specifying meter groups, and meters or participants can audit the group composition to detect whether they are tricked into participating in compromised groups. For this purpose a tamper evident log of group participants can be kept by the meters or the certified aggregates can be kept by users to prove any deviation from the genuine groups. Pragmatically energy providers are likely to be curious but unlikely to engage in behaviour that can be shown to deviate from their obligations, be it contractual or regulatory.

Individual customer may wish to opt-out of smart metering all together. Supporting regions with such customers is not a problem for the aggregation protocols but a challenge for our comparison protocols. Consider a single meter

within a region not participating in computing the privacy friendly aggregate that is also metered by the aggregate meter: the difference between two sum of participating readings and the aggregate meter will end up being the consumption of the meter that has opted out. This is perverse as it results in a privacy sensitive user being even more vulnerable by opting out than by participating in the protocol.

4.5 Support for Settlement, Profiling and Forecasting

The primary aim of the aggregation protocol is to detect whether the sum of meter readings corresponds, or at least is close to, the reading of an aggregate meter. This allows electricity distributors to detect whether any fraud might be taking place, in the case the sum of reported readings are substantially below what is reported by the aggregate meter. In this settling meter groups must correspond to the physical distribution network since there should be a correspondence between the computed aggregate and the metered aggregate.

Other processes in the energy industry rely on aggregate of readings, which do not have such a straight forward correspondence. We will concentrate on two particular processes, namely *settlement* and *profiling*, and discuss how our aggregation protocols could be used to solve them in a privacy friendly manner. For the purposes of the discussion we assume it is practical to extract the aggregate as from the protocols, and not merely to match it to a known consumption.

First we give an overview of *settlement* and *profiling* in the energy industry – both processes that are buried deep in the infrastructure:

Settlement. The UK energy market works by separating the supply of energy from its generation. A number of suppliers draft contracts with generators to produce a certain amount of electricity within a sequence of half-hourly time periods. Yet, the actual load of the network is monitored by the UK grid, that may also issue orders to increase or reduce generation in the short term to meet the actual demand. The settlement process determines whether the contracts of suppliers with generators covered the actual demand of their customers, or whether specific suppliers need to pay more for any extra generation, or under consumption. To determine whether the production of electricity for each supplier matched their demand an estimate of the total amount of electricity consumed by customers of each supplier has to be produced. We therefore discuss how our protocols could be used to supply such estimates.

Profiling. Both suppliers and national grids need data on which to base electricity models and forecasts. Short term forecasts are related to very short term demand and whether. Longer term forecasts depend on other factors including the effects new devices have on consumption, socio-economical profiles of users, different patterns of consumption per region or sector of the economy. When raw data is available an analysts can use them to train their models. In the absence of raw data volunteers are recruited or payed to construct profiles. We show that our protocols can be used to extract load profiles for different populations despite aggregation.

Trivial solutions. Both issues of settlement and profiling boil down to computing aggregates over different sets of meters. For settlement it would suffice to compute aggregates of meters associated with each distinct supplier to estimate the total energy consumption of their user base over time. This would be a far superior estimate than those produced by current methods (based on aggregate consumption and average profiles). A trivial solution for profiling would require meters to be groups according to the profile criteria: different temperatures, regions, socio-economic class, etc.

The trivial solution could work but might not be practical. For settlement, there is no uncertainty about the association of meter and supplier. Yet, changing the meter group requires expensive re-keying in all our protocols. Depending on how dynamic the energy market is this may happen multiple times every year. For profiling the task of grouping meters according to pre-determined categories is even harder. For example analysts may be interested in observing the effect temperature has on the energy consumption of a household over the winter holidays. Yet, it is not easy to predict the exact temperatures to group meters accordingly. Similarly, it is difficult to group meters by family size or composition of family, as demographics are subject to frequent change. In the case of socio-economic profiling, the data may simply not be available at an individual level to assign meters into groups – and further privacy concerns may arise if this is attempted.

Finally the trivial solution require meters groups to be tuned to extracting particular aggregates, or require them to output readings associated with multiple groups. Depending on the scheme used this increases computation and communication costs, while degrading the quality of privacy protection.

Inference on random population meter groups. Meters may be assigned to arbitrary groups, within which readings are aggregated, and yet and regression analysis can be applied to extract statistics from arbitrary meter populations. This approach decouples the assignment of meters into groups from any consideration of what statistics are to be extracted at a later time, alleviating the shortcomings of the trivial solution.

Consider a number N of meter groups $\mathcal{G}_i$ which run our protocols to calculate at each time period an aggregate of their consumption $S(\mathcal{G}_i)$. We denote as $\boldsymbol{S}$ the column $(N \times 1)$ matrix with elements $S(\mathcal{G}_i)$. An arbitrary partition of meters and a function $\mathcal{P}$ that is applied to each group $\mathcal{G}_i$ returns the number of meters $\mathcal{P}(\mathcal{G}_i)$ in the group within that partition. The domain of $\mathcal{P}(\mathcal{G}_i)$ is as expected $[0, |\mathcal{G}_i|]$.

The mean consumption of the meters within the partition $\mathcal{P}$ can be estimated from the aggregate readings $S(\mathcal{G}_i)$. We construct $\mathcal{M}$ a $N \times 2$ matrix with elements $\mathcal{P}(\mathcal{G}_i)$ and $|\mathcal{G}_i| - \mathcal{P}(\mathcal{G}_i)$, and compute:

$$\mathcal{R} = (\mathcal{M}^T \mathcal{M})^{-1} (\mathcal{M}^T \boldsymbol{S})$$

The 2×1 matrix $\mathcal{R}$ is the least squares estimator of the mean of the consumption of the population in $\mathcal{P}$ (in position 1×1) and the population of meters not in $\mathcal{P}$

(in position 2×1). This is a standard linear regression, and it can be extended to estimating mean consumptions of multiple partitions of meters simultaneously. Efficient techniques based on LU decompositions avoid the need for a matrix inversion in case multiple population partitions are required.

4.6 Converting a Comparison Protocol back into an Aggregation Protocol

The scheme as we described allows an aggregator to verify if an aggregate it already knows corresponds to the sum private measurement values it received. In many settings, however, an aggregator cannot measure the aggregated value - for example, a utility may be interested in the aggregate of the power output of all houses with photovoltaic energy generation, which are not connected to the same substation. Note that in this case the masking values do not cancel out – however, the aggregator can simply be provided with the sum of the masking values and thus effectively get the same effect.

While the comparison protocol supports fraud detection it requires reading from an aggregate meter. In some settings, such as gathering statistics, one may need to extract the sum of meter readings instead of comparing it to a known value.

A typical smart meter reading is a four byte value. If we assume up to 250 devices in one group, that would give us a 40 bit value for the aggregated reading. However, in most cases, the aggregator has a fairly good idea on the rough total consumption, as energy usage is fairly predictable - this would easily reduce the set of possible values into an area a normal computer can brute-force in a reasonable short time (Note that the brute force will only reveal the aggregate, while the individual contributions are still secure).

If the either the number of measurements of the measurement domain gets too big, the meters can easily split the measurement in a high- and low part and report both parts independently. The aggregator can then brute force both parts individually, reducing the computational effort on the backend to a level it can handle in a practical setting. The only setting in which this approach does not work is if the aggregation is performed over a large number of devices, e.g., a million meters. In this case, however, the entire protocol can be run independently on different subgroups of the devices without any loss of privacy.

5 Prototype implementations.

We implemented the *low-overhead* variant of the proposed scheme (described in Section 3.4) in the Python language. The code core with the cryptographic operations spans 89 lines of code. It uses the standard library hash function SHA-256, and a separate pure-python implementation of Curve25519 [21] for Diffie-Hellman key generation and derivation yielding 32 byte public keys. Readings and their cipher texts are represented using 4 bytes.

We tested our protocols in the setting of 100 meters reporting their aggregate consumption. Key generation took 0.013 s / meter and lead to 4790 bytes of total storage required for the 100 public keys and their associated meta-data. Key derivation, i.e. the computation of the secrets shared with other meters, took 1.371 s / meter. The 100 EC point multiplications using Curve25519 per meter dominate the cost of this operation. Each subsequent computations of the blinding factors required for obscuring readings took less than 0.001 s / meter. All reported figures are averages over 100 experiments.

The pure python implementation of Curve25529 is orders of magnitude slower than a native or optimised implementation, and dominates the cost of deriving shared keys. Such key derivation only happens when meter groups are formed, and can be amortised over an arbitrary period of time when groups are stable. The recurring cost of calculating blinding factors for readings take a negligible time as they only require the application of comparatively fast hash functions.

Implementation of regression techniques. The stability of meter groups can be maintained while extracting statistics about arbitrary partitions of the meters using the proposed regression based techniques. We partitioned a population of 1 million meters into 1000 groups of 1000 meters each reporting collectively their aggregated consumption. We then partitioned meters into two populations consuming electricity according to a population with different means μ_a and μ_b. We ensured that at least 50 meters from both populations are present in each meter group, and inferred the means μ_a and μ_b using our regression analysis.

The regression algorithm for inferring μ_a and μ_b took less than 0.001 seconds to run, and was implemented in 30 lines of pure python with standard numerical libraries. As expected it returns the values of the means with negligible error. (See [22] for a detailed treatment of error analysis in regression.) This demonstrates that computing statistics from aggregate measurements using regression analysis is computationally feasible even at a national scale.

6 Conclusion.

A naive way of implementing privacy-friendly aggregation and comparison protocols would involve a trusted party collecting all raw readings to aggregate them. This is indeed the approach currently discussed for the UK smart-metering deployment and others. We argue this is not necessary and present a family of protocols to achieve the same functionality without the need to ever disclose raw meter readings. Different protocols have different advantages we discuss, in terms of their properties, their cost, their deployment model, and how they interrelate with other smart-metering privacy technologies. Similar approaches could be extended to aggregates for other utilities as well as a general set of techniques to gather real time statistics without revealing private data.

Acknowledgements. We would like to thank Michael John for insightful comments on the reality of smart metering, and Lejla Batina and Jaap-Henk Hoep-

man, for helpful discussions and for taking the patience to read and comment on early versions of this papers.

References

1. European Parliament: DIRECTIVE 2009/72/EC (2009)
2. Cuijpers, C., Koops, B.J.: Het wetsvoorstel 'slimme meters': een privacytoets op basis van art. 8 evrm. Technical report, Tilburg University, oct. 2008. Report (in Dutch)
3. The Smart Grid Interoperability Panel Cyber Security Working Group: Smart Grid Cybersecurity Strategy and Requirements, US National Institute for Standards and Technology (NIST). http://csrc.nist.gov/publications/nistir/ir7628/nistir-7628_vol2.pdf (2010)
4. Garcia, F.D., Jacobs, B.: Privacy-friendly energy-metering via homomorphic encryption. In: 6th Workshop on Security and Trust Management (STM). (2010)
5. Molina-Markham, A., Shenoy, P., Fu, K., Cecchet, E., Irwin, D.: Private memoirs of a smart meter. In: 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings (BuildSys 2010), Zurich, Switzerland (November 2010)
6. Rial, A., Danezis, G.: Privacy-preserving smart metering. Technical Report MSR-TR-2010-150, Microsoft Research (November 2010)
7. Danezis, G., Kohlweiss, M., Rial, A.: Differentially private billing with rebates. Technical Report MSR-TR-2011-10, Microsoft Research (February 2011)
8. K. Kursawe: Some Ideas on Privacy Preserving Meter Aggregation. Technical Report ICIS–R11002, Radboud University Nijmegen (February 2011)
9. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security. (1993) 62–73
10. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. J. Cryptology **1**(1) (1988) 65–75
11. Hao, F., Zielinski, P.: A 2-round anonymous veto protocol. In Christianson, B., Crispo, B., Malcolm, J.A., Roe, M., eds.: Security Protocols Workshop. Volume 5087 of Lecture Notes in Computer Science., Springer (2006) 202–211
12. Golle, P., Juels, A.: Dining cryptographers revisited. In Cachin, C., Camenisch, J., eds.: EUROCRYPT. Volume 3027 of Lecture Notes in Computer Science., Springer (2004) 456–473
13. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In Biham, E., ed.: EUROCRYPT. Volume 2656 of Lecture Notes in Computer Science., Springer (2003) 255–271
14. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In Feigenbaum, J., ed.: CRYPTO. Volume 576 of Lecture Notes in Computer Science., Springer (1991) 129–140
15. Schnorr, C.: Efficient signature generation for smart cards. Journal of Cryptology **4**(3) (1991) 239–252
16. Chaum, D., Pedersen, T.: Wallet databases with observers. In: CRYPTO '92. Volume 740 of LNCS. (1993) 89–105
17. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In Odlyzko, A., ed.: CRYPTO. Volume 263 of LNCS., Springer (1986) 186–194

18. Camenisch, J., Stadler, M.: Proof systems for general statements about discrete logarithms. Technical Report TR 260, Institute for Theoretical Computer Science, ETH Zürich (March 1997)
19. Diffie, W., van Oorschot, P.C., Wiener, M.J.: Authentication and authenticated key exchanges. Des. Codes Cryptography **2**(2) (1992) 107–125
20. Borisov, N., Goldberg, I., Brewer, E.A.: Off-the-record communication, or, why not to use pgp. In Atluri, V., Syverson, P.F., di Vimercati, S.D.C., eds.: WPES, ACM (2004) 77–84
21. Bernstein, D.J.: Curve25519: New diffie-hellman speed records. In Yung, M., Dodis, Y., Kiayias, A., Malkin, T., eds.: Public Key Cryptography. Volume 3958 of Lecture Notes in Computer Science., Springer (2006) 207–228
22. Gelman, A., Hill, J.: Data Analysis Using Regression and Multilevel/Hierarchical Models. 1 edn. Cambridge University Press (December 2006)

A Proof of Basic Comparison Protocol

We will demonstrate protocol security of the basic comparison protocol, with random but round independent masking x_j under the Decisional Diffie-Hellman assumption in the Random Oracle Model [9].

Proof outline. We will proof correctness in the ideal world/real world model, i.e., define an ideal world setting (in which security is obviously given), and proof indistinguishability from the real world setting. Thus, we construct a simulator that gives the aggregator either data that is equal to the data generated in a real run, or equal to the data generated in the idealised one, and proof that the aggregator cannot tell the difference between the two.

This allows us to use a diagonalisation argument to argue that if an attacker cannot tell where the switch from ideal world to real world happens, she also cannot distinguish a fully ideal world from a fully real one. Now taking the later case and $k = 2$, we show that it is not

Attack model. Assuming the blinding-keys are generated and distributed securely, the end-user does not need to trust either the meter or the aggregator at all (in terms of privacy protection). The protocol itself is completely deterministic with no secrets that an end-user would not be allowed to know, so no information can be hidden inside the messages. It is not even necessary that the meter does the calculation itself in the first place - given the meter reading, an external device (e.g., an internet connected PC) could perform this task as well.

Similary, the aggregator only needs to assume that his deblinding key is proper to guarantee fraud prevention - the only fraud still possible is if two meters collude in a way that one meter overreports be the same amount another one underreports [5]. We do assume some security in the meter that assure that the values reported to the fraud detection are the same reported to the billing system, and that messages from the meter are authenticated (alternatively, if H

[5] There are scenarios, especially with variable tarrifs where that actually may make sense, but we safely can assume this to not be an issue for now

is a keyed hash function, it is sufficient for the meters to keep the corresponding key private). In this, we assume that attacks on the meter from the customer are usually done by circumventing the meter, rather than reprogramming the entire unit. This is a necessary assumption for any fraud detection, as we need to assure that the values the detection system gets are in some way related to reality; in the future work section, we direct towards a solutioon that would also allow completely hacked meters to be included.

While it is easy for an individual meter to cause false alarms – and in this, run some form of denial of service attack – this is not an issue for our protocol. As the whole point is to trigger an alarm if something goes wrong, and a certiofied meter launching a denial of service attrack would certainly qualifty as such, the protocol will act exactly as desired.

Note that in a practical setting, we can assume that the aggregator will not behave completely dishonest, but more what can be described as "flawed but non-criminal"; that is, data that is or can easily me made available will be abused, but the aggregator will not commit easy to detect criminal acts (e.g., invent hundreds of non-exsiting meters in the setup phase) to be able to spy on an individual meter; this will make the real-world key- and device management much easier.

Extra care has to be taken as the measurement values of the meters may come from a very restricted domain, and thus can easily be predicted in a realistic setting.

Notations We denote with n the number of honest meters. We assume $n \geq 2$, which is the minimum required for any aggregation. In addition to the n honest meters, we allow for an unlimited number of dishonest meters. As there is no communication between meters, the dishonest meters play no real role in the protocol or the proof. We call m the number of measurements. There is no limit on m, apart from m being polynomial in the security parameter.

Let G be an appropriate group for Diffie Hellman; the following variables are elements in G:

$x_{i,j}$ = blinding value for measurement i on meter j

$c_{i,j}$ = measurement value for measurement i on meter j.

In addition, we have a hash function $H : (\{0,1\}^* \to G$. We assume H to have random oracle properties. For readability, we define $g_i = H(i)$. Note that the domain for the $c_{i,j}$ can be small and predictable, i.e., an attacker can brute–force $c_{i,j}$ given g and $g^{c_{i,j}}$.

DDH For the simulation, we have a given instance of the Decision Diffie Hellman problem, i.e., we have given $g, h_1 = g^a, h_2 = g^b, h_3 \in G$ and need to decide if $h_3 = g^{ab}$.

The Ideal and the Real world We first define an idealised protocol, in which privacy is assured in an information theoretical sense. In this idealised world, every measurement i at meter j has a unique, independent blinding value $x_{i,j}$ such that for all i, $\sum_j x_{i,j} = 0$.

For measurement i, meter j sends $m_{i,j} = x_{i,j} + c_{i,j}$ to the aggregator.

This is information theoretically secure (For everything we send, there are blinding values for all possible measurements that could have led there). We may need to be a little careful with the distribution, as the $c_{i,j}$ are poorly distributed.

If we now choose a (public) generator g_i of an appropriate group G, sending instead

$$g_i{}^{x_{i,j}+c_{i,j}} \ ,$$

is at least as secure as sending $m_{i,j}$ directly. This is our ideal scheme.

Recall that $H : \{0,1\}^* \to G$ is a hash-function with random oracle properties. We call $H(i) = g_i$.

In the real world, we have $x_{i,j} = x_{i',j}$ for all i, i', i.e., a given meter uses the same blinding values for all measurements. In this case, we also denote $x_{i,j}$ as x_j. Let x_j be the blinding value for meter j, and $c_{i,j}$ the measurement i for meter j. Thus, for measurement i, meter j sends

$$g_i{}^{x^j+c_{i,j}} \ .$$

The Simulation We will now construct a reduction that will use an adversary which can distinguish the ideal from real world protocol to solve DDH.

To this end, we introduce (ℓ, k)-hybrides $\ell < n$ and $k \leq m+1$, and define that Meters $1, ..., \ell - 1$ behave ideal. Meters $\ell + 1, ..., n$ behave real. Meter ℓ behaves

- ideal for measurements $1, \ldots, k-1$
- real for measurements $k, \ldots, m$.

Note that an $(\ell, m+1)$-hybrid behaves exactly the same as a $(\ell+1, 1)$-hybrid and that it is not possible to distinguish between $(1, k)$-hybrides, as

$$g_i^{x_{i,1}} = \frac{1}{\prod_{j=2}^{n} g_i^{x_j}} \ .$$

The randomness of $x_{i,1}$ is fixed to a unique value by the sum-constraint and the behavior of the other meters.

We prove that adjacent hybrids for $j > 1$ cannot be distinguished under the DDH assumption: We first set $g_k = H(k) = h_1 = g^a$; this is where the random oracle property of H is required.

As the next step, we want to set $x_\ell = b$, even though the simulator only knows $h_2 = g^b$. We know that the first meter behaves ideal. All meters can behave following the description of the hybrid as is, and the first meter uses

$$g_i^{x_{i,1}} = \frac{1}{\prod_{j=2}^{n} g_i^{x_{i,j}}}$$

Note that $g_k^{x_{k,j}} = h_3$.

Now, if $h_3 = g^{ab}$, meter ℓ sends $g^{ab} = {g_k}^b = {g_k}^{x_\ell}$ as its blinding value, i.e., meter ℓ behaves real for measurement k. Else, the blinding value it uses is random, and thus the meter behaves ideal for measurement k. Therefore, we have the following lemma:

Lemma 1. *Given above construction, any attacker that can distinguish whether meter ℓ behaves real or ideal for measurement k, can also solve DDH.*

Given this lemma, we can now use a diagonalisation argument to argue that full real behaviour is indistinguishable from full ideal behaviour. Suppose we have an attacker that can distinguish our real- from our ideal world setting with some advantage ϵ. We then provide that attacker with all our intermediate steps, where some meters/measurements behave real and the others behave ideal. This means there is some setup where the one individual measurement is decisive, i.e., the attacker will tend towards 'ideal' if that measurement is ideal, and towards 'real' otherwise. This is the setting where we can use our above simulator to turn it into a DDH decider.