PRIVACY-PRESERVING SMART METERING WITHOUT A TRUSTED-THIRD-PARTY

Tobias Jeske
Institute for Security in Distributed Applications, Technical University of Hamburg, 21079 Hamburg, Germany

Keywords: Smart metering, Smart grid, Cryptography, Zero-knowledge protocols, Privacy.

Abstract: Smart meters report the current electricity consumption over the internet back to their energy providers. Finely-sampled power consumption enables the energy provider to learn the habits of the customer's household in which the smart meter is installed. This paper presents a protocol which preserves customer privacy but also allows the detection of unregistered smart meters and prevents spamming and replay attacks. A trusted-third-party is not needed. This protocol, whose security proof relies on the strong RSA assumption and the random oracle model, is based on zero-knowledge techniques. The protocol has been implemented on different hardware platforms and benchmark results are given.

1 INTRODUCTION

For energy providers it is important to know the pattern of their customers' daily electricity consumption. A difference between supply and demand can impinge on power frequency and voltage (Müller, 2010). Today energy providers plan electricity demands on the basis of statistical data based on the knowledge of past power consumption and on the experience of how power consumption will vary in the future (e.g. seasonal variations, upcoming mega-events).

Electricity demands fluctuate over a day and are generally at their highest in the morning and evening. An energy provider uses different sources to supply electricity. Base load power stations (nuclear power plants, lignite-fired power plants) are characterized by a nearly constant energy transfer running the whole day, offering electricity at a low price. The investment in new nuclear power plants is high, whereas the costs of operation are comparatively low. For this reason, it is economically advantageous to continuously operate them at maximum power. Lignite-fired power plants usually have long start-up times, which also make them unsuitable to compensate peaks in the power demand. Instead, peak-load electricity generation plants like pumped-storage/gas turbine power stations are used. They have short start-up times, are more flexible and provide energy whenever needed but at higher costs. If the demand is too high or the energy provider does not have any power plants, electricity is bought at the electricity stock exchange.

Different electricity sources marked by varying prices and fluctuating demands necessitate a sophisticated resource planning. Renewable energies like wind and solar power make the situation even more difficult. Energy generation depends on the weather; in some countries like Germany energy providers are obliged to feed renewable energy into the electricity grid (German Federal Ministry of Justice, 2008).

Energy providers have a huge interest in precisely predicting their customers' electric energy consumption in order to be always able to choose the economically most advantageous energy source. Knowing the current electricity demands gives more planning stability and permits differentiated price models. Energy providers couple the costs for electric energy generation to the demands, for example by charging higher fees during peak hours, and give customers incentives to save energy and money (Müller, 2010) (McDaniel and McLaughlin, 2009).

1.1 Smart Meter & the Smart Grid

Smart meters can report current electricity consumption back to their energy providers (e.g. every 15 minutes) (Müller, K. J. et al., 2011). Berg Insight estimates that 302.5 million smart meters will be

Jeske T..
PRIVACY-PRESERVING SMART METERING WITHOUT A TRUSTED-THIRD-PARTY.
DOI: 10.5220/0003513001140123
In *Proceedings of the International Conference on Security and Cryptography* (SECRYPT-2011), pages 114-123
ISBN: 978-989-8425-71-3

installed worldwide in 2015 (Berg Insight, 2010). In Germany, the Energiewirtschaftsgesetz has made smart meters compulsory in new and renovated private houses (German Federal Ministry of Justice, 2005).

Modern smart meters visualize the power consumption in real-time and present it to the customers. The customer is notified when the price per kW/h is attractive, which enables shifting usage hours of electrical equipment (e.g. air conditioning) to non-peak hours.

The Smart Grid integrates all actors in the electricity market into a complete system, in which the producer, storage and consumer of electrical energy communicate with each other. Smart meters are important devices in the Smart Grid concept (McDaniel and McLaughlin, 2009). They allow to control and communicate with electrical devices in the household (the smart meter could start the washing machine in a specific time interval when the kW/h price is low) and to predict power consumption. A running washing machine, for example, could inform the smart meter of its power consumption for the wash cycle. In this case the smart meter would forward this information to the energy provider to optimize the resource planning.

1.2 Smart Meter & the Privacy Problem

Sending finely-sampled power consumption data to the energy provider allows the energy provider to learn the habits of the consumer's household in which the smart meter is installed. The electrical equipment of a household has a characteristic energy profile easy to identify (Müller, 2010).

Figure 1 (left) represents the energy profile of a baking oven marked by a noticeable high energy consumption of about 1800 W. At the beginning, during the heating-up period, the oven is heated from room temperature to the target temperature. Then the heating element is switched on from time to time to keep up the temperature. This power consumption pattern is typical and characteristic of most electric baking ovens.

If one superimposes the consumption of all electrical equipment, one can still recognize a baking oven because of its high individual energy consumption. Even if a customer employs more devices consuming much electric energy (e.g. an air conditioner), different devices can be identified due to usage time and curve progression. A fridge is usually the only high power consumption device running over night. Figure 1 (right) shows the electrical energy consumption of an entire day. Despite the fact that the resolution is only four samples per hour, electrical equipment, for instance a baking oven and a dishwasher, can be detected in the chart.

What is more difficult but not impossible is the identification of electrical equipment with low power consumption. Even in this case, usage time and the fact that most of the electrical equipment is not continuously switched on makes it possible to identify individual devices. Filtering out already identified devices improves the identification of the remaining ones.
Müller (Müller, 2010) gives the following criteria to identify electronic equipment:

1. Power consumption
2. Load profile of different device types
3. Typical operating cycle
4. Time of usage
5. Frequency of usage

If someone evaluates the energy profiles, it is for instance easy to see when someone is at home (light, TV etc. is switched on). This information is quite useful for burglars. Questions regarding living habits like "How many times do the household members cook?" or "When does the customer go to the toilet at night?" (light is switched on!) are not private any longer.

In summary, on the one hand, the energy provider benefits from knowledge of a detailed energy profile to improve the energy consumption forecast. On the other hand, customers do not want energy providers to gain insight into their habits. In this paper we present a new solution to solve these two (conflicting) requirements.

The paper is structured as follows: In section 2 we present our scenario and aims for privacy-preserving smart metering. In section 3 we explain the main concept as well as some general ideas behind our solution. In section 4 we present a technical description. Implementation results are presented in section 5. In section 6 we cover related work before drawing a conclusion in section 7.

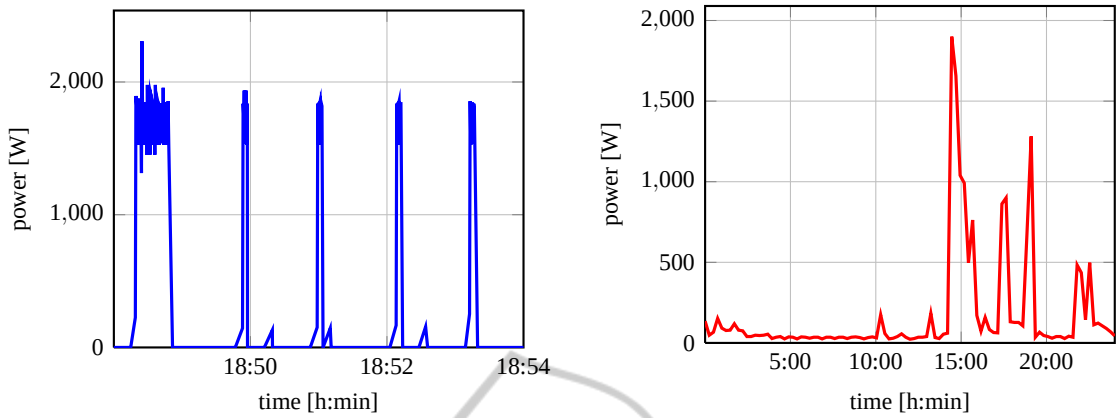

Figure 1: Electrical power consumption of a baking oven (left) and a single household over 24h (right).

2 SCENARIO & AIMS FOR PRIVACY-PRESERVING SMART METERING

In our scenario, every household has one smart meter communicating with one energy provider. If a household has several metering devices, a MUC (Multi Utility Communication) controller is usually used as a gateway. In our protocol the MUC device acts as the smart meter for simplification. The meters and MUC of a household build a trusted environment between each other.

Every smart meter communicates with the energy provider over a standard network protocol like TCP/IP. We require that energy providers cannot identify smart meters by the sender address (e.g. IP or MAC address). In section 5 we discuss how this requirement can be fulfilled in real-world implementations.

Other solutions usually require a trusted-third-party (TTP) to manage membership (Petrlic, 2011) (Garcia and Jacobs, 2010). We take the view that a TTP is difficult to realise in practice. First, energy providers want to have the flexibility to add and remove smart meters from their network without contacting a TTP. Secondly, we believe that customers who do not trust the energy provider have no motivation to trust a TTP (e.g. run by the government) instead.

We require that smart meters can periodically (e.g. every 15 minutes) send their power consumption to the energy provider for load reporting. The customer's privacy must be preserved.
The smart meter accumulates the energy consumption and submits it to the energy provider for billing purposes (e.g. every month). We assume that the smart meter regularly downloads a billing policy from the energy provider in an authenticated way and charges the energy consumption against the policy. During the billing process, the customer must be identified.

Smart meter authentication is important in every protocol stage. Energy providers must be able to verify if the consumption data was sent from a valid smart meter that is part of the provider's grid.

In summary, here is a list of the aims we want to achieve with our solution:

1. *Anonymity*
   The privacy of every customer is preserved. It is impossible for the energy provider or anybody else to get information about the customer's living habits.
2. *Unlinkability*
   Data packets that are sent from the same smart meter at different times for the purpose of load reporting cannot be linked to each other.
3. *Misauthentication Resistance*
   Energy providers can detect unregistered smart meters and only accept data sent from smart meters being part of the provider's network.
4. *Invoicing*
   Smart meters send the energy provider cost calculations containing the power consumption for a certain time interval. The energy provider can identify smart meters for invoicing.
5. *Revocability*
   It is possible to remove smart meters from the provider's network. If an energy provider withdraws from the contract concluded with the cus-

tomer, data packets coming from the smart meter must be rejected.

6. *Prevention of Spamming and Replay Attacks*
   Malicious customers are not able to spam the energy provider with faked data and consequently spoil the power consumption forecast. Replay attacks are prevented. Each data packet which contains measurement data from a unique smart meter and which covers a unique time interval can be submitted only once to the energy provider. This prevents replay attacks.
7. *No Delay*
   It is important for load reporting that smart meters submit consumption data to the energy provider without any delay. We require a maximum delay of 15 minutes.
8. *Data Quality / Correctness*
   The energy provider must be assured that the smart meter correctly sends the consumption data.
9. *No TTP / Practicability*
   A protocol should not require a TTP for key or membership management. The energy provider can add (or remove) smart meters from the grid without the help of another party.

3 MAIN CONCEPT

In this section we explain the main concept behind our solution to achieve all aims listed in section 2. A detailed and more technical explanation follows in the next section.

Our protocol consists of two sub-protocols, the *invoicing* and the *load reporting* protocol.
A smart meter ($\mathcal{S}$) uses the *invoicing protocol* to send the accumulated electric power consumption to the energy provider ($\mathcal{P}$). Basically, $\mathcal{S}$ uses standard, asymmetric key cryptography to sign and encrypt the invoicing data. The identity of the customer and the corresponding smart meter is revealed (aim 4).
In general, the *load reporting* protocol is based on the idea of group signature schemes. $\mathcal{S}$ is a member of a group of smart meters with $\mathcal{P}$ as the group manager. Using group signature schemes, $\mathcal{S}$ signs the consumption data for load reporting in the name of the group. $\mathcal{P}$ can only verify group membership. Group signature schemes allow us to separate valid smart meters belonging to the energy provider's grid from other (malicious) smart meters (aim 3). $\mathcal{S}$ stays anonymous. Even if $\mathcal{S}$ signs two (different) messages, it is practically impossible for $\mathcal{P}$ to find out that both signatures come from the same smart meter. Using a group signature scheme, we achieve the first two aims, anonymity and unlinkability.

During the invoicing protocol, $\mathcal{S}$ is added to the group because $\mathcal{S}$ is identified. $\mathcal{S}$ creates a ticket and $\mathcal{P}$ signs it without knowing it. Later, $\mathcal{S}$ uses signature and ticket to prove group membership during submission of anonymous load reporting data. We use zero-knowledge protocols in order not to reveal signature and ticket and to preserve $\mathcal{S}$'s privacy. Every ticket can be used only once to prevent replay attacks (aim 6). The signature for the secret ticket contains a time stamp, which allows us to diminish the threat of spamming attacks (aim 6).

4 PROTOCOL DETAILS

As explained in the previous section, we need a group signature scheme for our protocol. Our solution is based on the Camenisch and Lysyanskaya (CL) signature scheme because of its efficiency (Camenisch and Lysyanskaya, 2003). Its security relies on the strong RSA assumption. We use the version of Camenisch and Groth to improve performance (Camenisch and Groth, 2005).

The CL signature scheme consists of two protocols. Using the first protocol, a signer issues a signature on a committed value. In the second protocol, the party knowing the committed value can prove knowledge of a signature by using a signature proof of knowledge.

This paper follows the notation of Camenisch and Stadler (Camenisch and Stadler, 1997) to describe zero-knowledge proofs of knowledge.

4.1 Setup

Every smart meter $\mathcal{S}$ has a long-term secret key $\mathcal{S}_{LTSK}$ and long-term public key $\mathcal{S}_{LTPK}$. $\mathcal{S}_{LTSK}$ serves as $\mathcal{S}$'s signing key and $\mathcal{S}_{LTPK}$ is the corresponding verification key. The energy provider $\mathcal{P}$ also has a long-term key pair $\mathcal{P}_{LTSK}$ and $\mathcal{P}_{LTPK}$, which is used for sending encrypted data to $\mathcal{P}$. The key pairs are created once by the corresponding parties. We assume that $\mathcal{S}_{LTPK}$ unambiguously identifies $\mathcal{S}$.

In our protocol $n = p \cdot q$ represents a RSA modulus of bit length l_n, where p and q are two large safe prime numbers only known to $\mathcal{P}$. $\mathcal{P}$ randomly chooses $a, b, d, f_1, f_2, g, h \in_R QR_n$, where QR_n denotes the set of quadratic residues modulo n. It also sets $TS_{log} \in \mathbb{Z}_n$, a time delay in which $\mathcal{S}$ cannot

submit data to $\mathcal{P}$.

When the customer concludes a contract with a new energy provider $\mathcal{P}$, $\mathcal{S}$ establishes a secure and authenticated connection with $\mathcal{P}$. This could e.g. be realised using SSL/TLS where $\mathcal{S}$ has pre-installed root certificates to verify $\mathcal{P}$'s certificate or alternatively public key.
$\mathcal{S}$ submits the meter number together with $\mathcal{S}$'s public key to $\mathcal{P}$ for registration. To identify the customer $\mathcal{S}$ could also send $\mathcal{P}$ a PIN code over the secure channel. The customer could e.g. receive the PIN from $\mathcal{P}$ via a letter.

4.2 Invoicing

$\mathcal{S}$ runs the invoicing protocol to send $\mathcal{P}$ the cost calculation comprising the power consumption for a certain interval. At the end of the protocol, $\mathcal{S}$ has not only sent $\mathcal{P}$ the accumulated power consumption, but the smart meter is also in possession of a signature for a secret ticket x which allows $\mathcal{S}$ to anonymously send $\mathcal{P}$ load reporting data (figure 2). The invoicing protocol corresponds to the first protocol of the CL signature scheme.

The protocol works as follows. First, $\mathcal{S}$ randomly chooses a secret number x, the ticket. The ticket is embedded in a Groth commitment C_{x_1} (see appendix 7) and the power consumption is accumulated in $data_{invoicing}$. $\mathcal{S}$ creates a signature S_1 with its long-term private key $\mathcal{S}_{LTSK}$ for values C_{x_1} and $data_{invoicing}$. Afterwards $\mathcal{S}$ sends C_{x_1}, S_1, $data_{invoicing}$ and its public key (as an identifier) to $\mathcal{P}$. The consumption data $data_{invoicing}$ and $\mathcal{S}_{LTPK}$ are encrypted under $\mathcal{P}$'s public key. If $\mathcal{S}$ is a valid smart meter and part of $\mathcal{P}$'s grid, $\mathcal{P}$ accepts the consumption data and otherwise rejects them (aim 3, aim 5). If $\mathcal{S}$ is a valid smart meter, $\mathcal{P}$ creates a signature for $\mathcal{S}$'s secret ticket x.

The signature creation and expiration time TS_p and TS_e are embedded in the signature. After $\mathcal{P}$ has sent the signature parameters (r', e, v, TS_P, TS_e) back to $\mathcal{S}$, $\mathcal{S}$ verifies the signature (s, e, v, TS_P, TS_e) and checks the time stamps. It is important that TS_e is coarse enough (e.g. years or months) so that $\mathcal{P}$ cannot identify $\mathcal{S}$ by choosing a unique time stamp for $\mathcal{S}$.

4.3 Submitting Consumption Data for Load Reporting

Using the load reporting protocol, $\mathcal{S}$ can anonymously send $\mathcal{P}$ the energy consumption of a certain interval. The protocol is shown in figure 3. The second protocol of the CL signature scheme is used as a basic building block.

First, $\mathcal{S}$ proves in zero-knowledge that it is in possession of a valid ticket x, i.e. that $\mathcal{S}$ has a valid signature (s, e, v, TS_P, TS_e) from $\mathcal{P}$ for ticket x.

Second, $\mathcal{S}$ proves in zero-knowledge that TS, the time stamp of the measured power consumption $data_{reporting}$, is greater than $TS_P + TS_{log}$. If this is not the case, $\mathcal{S}$ submits data too soon and $\mathcal{P}$ rejects the submission. An anonymous submission is only allowed every TS_{log} minutes.

The considerations mentioned above lead with $C_v = vb^{\tilde{r}}$, $\tilde{r} \in_R (0, 2^{l_n/2})$ to the following signature proof of knowledge (SPK):

$$\mathsf{SPK}\left[\begin{array}{l}(\varepsilon, \sigma, \gamma): \\ d = C_v^{2^{l_E}+\varepsilon} a^{-x} b^{\sigma} f_1{}^{-4\gamma} f_2{}^{-TS_e} \wedge \\ TS \geq TS_p + TS_{log}\end{array}\right](m) \quad (1)$$

Every ticket x can be used only once to prevent replay attacks. $\mathcal{S}$ reveals x and TS_e in clear text to $\mathcal{P}$, who puts them on a blacklist BL. To limit the blacklist size, $\mathcal{P}$ removes expired tickets from the blacklist from time to time.

The message m in proof (1) contains the measurement data $data_{reporting}$ and a commitment $C'_x = a^{x'} b'^{r_1}$ for x', the ticket for the next submission step. If $\mathcal{S}$ has followed the protocol and is a valid smart meter, $\mathcal{P}$ creates a signature for x' like in the *invoicing* protocol. $\mathcal{S}$ can use ticket x' to submit consumption data anonymously after $TS'_P + TS_{log}$ minutes.

4.4 Security Analysis

In our security analysis we consider three players, a trusted $\mathcal{S}$, $\mathcal{P}$ and an untrusted $\mathcal{S}$. We here want to discuss how our solution resists common attacks in a smart metering environment.
The security of the underlying CL signature scheme and our zero-knowledge proofs is based on the RSA assumption and the random oracle model. We refer the reader to (Camenisch and Groth, 2005) and (Groth, 2005) for more information.

4.4.1 Trusted Smart Meter

Customers manipulate their smart meter to forge the power consumption data for invoicing (electricity theft) and load reporting:

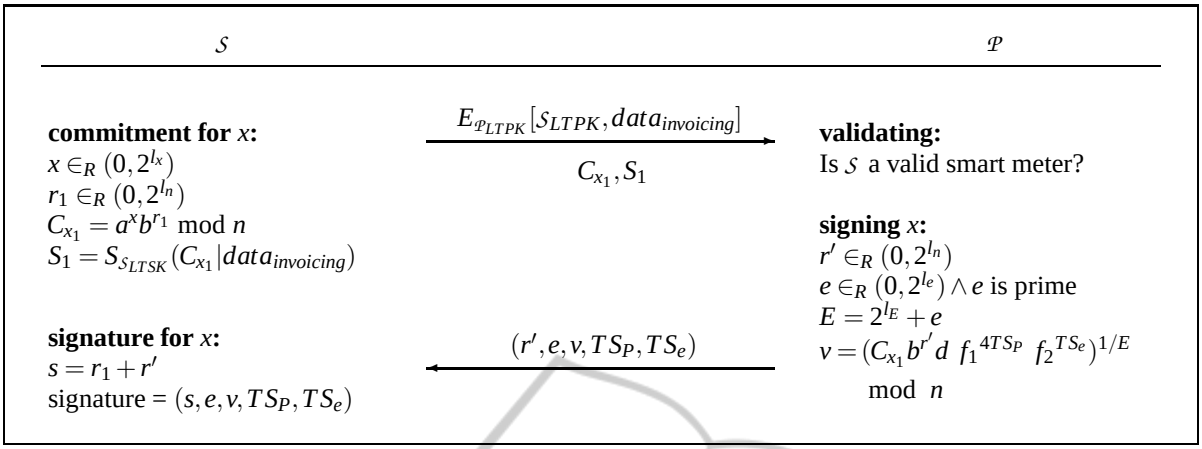

Figure 2: $\mathcal{S}$ submits to $\mathcal{P}$ the accumulated power consumption for invoicing.

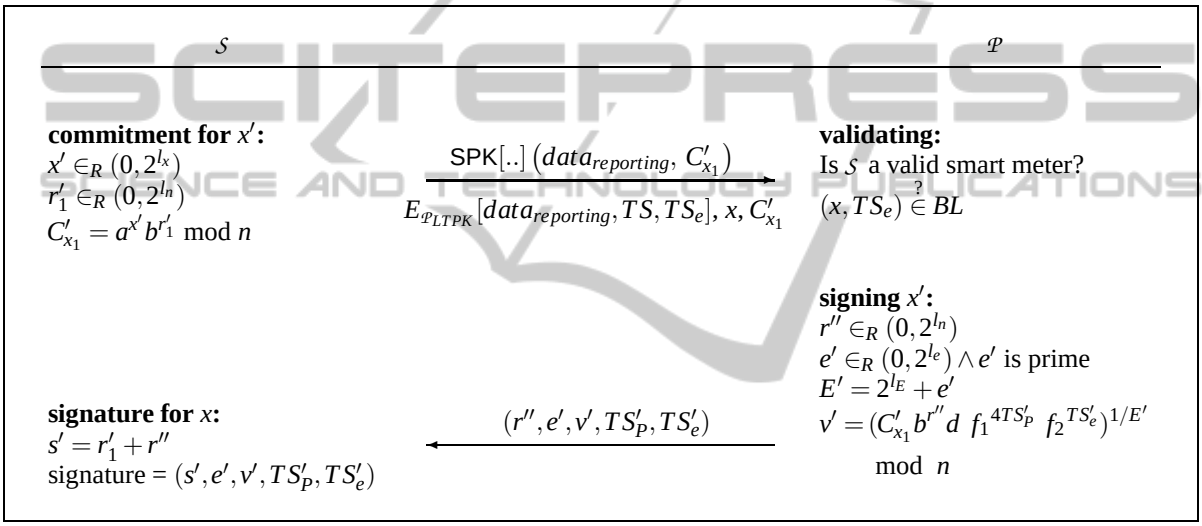

Figure 3: $\mathcal{S}$ sends to $\mathcal{P}$ consumption data anonymously for load reporting.

Electricity theft is prevented by putting the smart meter under seal. A trusted-platform-module (TPM) detects (malicious) modifications in the smart meter software and protects the key material (Petrlic, 2011) (Lemay et al., 2007). Otherwise, Lemay et al. note the high power consumption of a TPM in idle mode. In our protocol we do not need such a fully-functional TPM. Instead, we only require that the measured consumption data can never be manipulated before signature S_1 is calculated. The smart meter must compute S_1 in a trusted environment (aim 8).

To prevent counterfeiting of consumption data during the load reporting protocol (e.g. to inflict damage on the consumption forecasting), the hash function H must operate in a trusted environment and hashes the measured consumption data before it leaves the trusted environment. A malicious customer has to follow the protocol to persuade the energy provider to accept the consumption data. This is a consequence of the proof-of-knowledge property (see appendix 7).

4.4.2 Energy Provider

The energy provider acts maliciously by trying to create a finely-grained energy consumption profile of the customer. Our solution prevents this because of the underlying group signature scheme. The monthly submission of the aggregated energy consumption is too coarse to create profiles.

4.4.3 Untrusted Smart Meter

The main motivation of an attacker with an untrusted smart meter (e.g. a malicious competitor of the energy provider) is to become part of the energy provider's grid to perturb the energy consumption forecast. We assume that the attacker opens the smart meter or constructs a meter. A trusted environment no longer exists. The attacker can send the energy provider arbi-

trary data but cannot create a valid signature for it. To become a member of the grid, he must either have a valid public / private key pair registered at the energy provider or he must prove in zero-knowledge that he has a valid signature for a ticket not yet consumed. Even if the attacker has a public / private key pair, the smart meter can only anonymously submit consumption data every TS_{log} minutes, which reduces the impact of malicious data on the forecast (aim 3, aim 6).

5 IMPLEMENTATION & BENCHMARK

We have implemented our solution on three different hardware platforms. To measure performance on the energy provider's side, we run our protocol on a standard server machine (Intel Xeon X3460 @2.8 GHz, 8 GB RAM, Windows Server 2008 R2 Enterprise). On the smart meter's side, we do not have a programmable smart meter available. Instead, we evaluate our solution on a Linksys WRT54GS wireless access point. The WRT54GS has a Broadcom BCM4712 processor running at 200 MHz. The router is equipped with 32 MB RAM and runs OpenWRT 10.03. We are of the opinion that this hardware comes close to real smart metering devices like the ZDUE-DSL MUC from Dr. Neuhaus. It has a 210 MHz Atmel AT91SAM9260 processor, 32 MB RAM and runs an embedded Linux. Low power consumption of smart meters is important. According to the specification (Dr. Neuhaus, 2010), the ZDUE-DSL MUC has a power consumption of typically 2.2 Watt where the Linksys router has a power consumption of 7.6 Watt idle and $<$ 8 Watt at full load.
We also run our protocol on a modified Seagate FreeAgent Dockstar (Marvell Kirkwood 1.2 GHz, 128 MB RAM, Debian Linux, Kernel 2.6.32), which has a better performance / power consumption ratio (3.6 Watt idle, $<$ 5 Watt at full load). The power consumption was measured using the Voltcraft Energy Logger 4000.

We set the bit length of the RSA modulus to $l_n = 2048$. The MPIR 2.2.1 bignum library (a fork of GMPlib) is used to implement multi-precision integer operations (Gladman, B. et al., 2010). If packets must be encrypted or signed with the long-term keys, we use RSA. SHA-1 characterized by a hash size of 160 bit is employed as the hash function H. Tickets are chosen in the interval $(0, 2^{l_x})$ with $l_x = 80$. The time lag interval is $TS_{log} = 10$ minutes. We randomly choose the signature creation time TS_P in the interval $[0, 44640)$ (one month) and randomly set the time stamp TS to $TS = TS_P + TS_{log} + [0, 15]$. The expiration time stamp TS_e is set to $TS_e = 201201$ (concatenation of year and month). Other protocol parameters are chosen regarding (Camenisch and Groth, 2005) ($l_E = 404$, $l_e = 60$, $l = 60$). We run our protocol 100 times and determine the average runtime.

Table 1 shows the benchmark results for $\mathcal{S}$ and $\mathcal{P}$ at each protocol stage. Unsurprisingly, the protocol performs better on a PC than on the embedded platforms. The results show that the protocol presented in this paper is practical on real smart metering devices. Even on low-performance hardware the protocol performs fast enough to submit the consumption data to the energy provider in a 15 minute window (aim 7). The protocol calculations of $\mathcal{P}$ are not time-consuming if a typical server machine is used. We have tested our design only on a single core. If e.g. all cores are used on the Intel Xeon X3460 CPU, the performance would be eight times higher.

In real-world implementations one has to deal with the problem that the smart meter can be (theoretically) identified by the underlying communication protocols (e.g. IP / MAC address). The identity of the smart meter is preserved but it is possible to link packets coming from the same smart meter (and the same sender's address).
If the smart meter is connected with the internet over DSL or GPRS, a connection reset can solve the problem. The smart meter gets a new IP address and new consumption data submissions are unlinkable to former ones. On the one hand, if the smart meter shares the internet connection, it is impractical to get a new IP address before every submission. On the other hand, internet providers usually automatically reset the connection after 24 hours. A compromise could be that the DSL connection is reset if there is no traffic on the DSL connection.
If the smart meter communicates over a GPRS connection using a public provider, a new connection will be established for every new submission. This also implicates a new IP address.
The Tor network (The Tor Project, 2011) can also be used to obfuscate the sender's address. Traffic is routed in an encrypted way over intermediate nodes.
Although we have to shift some trust to the internet service provider or intermediate nodes we don't need a TTP for key or group management (aim 9).

Table 1: Performance comparison.

	invoicing ($\mathcal{S}$)	**invoicing** ($\mathcal{P}$)	**reporting** ($\mathcal{S}$)	**reporting** ($\mathcal{P}$)
Server (Xeon X3460)	21 ms	19 ms	88 ms	65 ms
Dockstar (Marvell Kirkwood)	352 ms	324 ms	1469 ms	1080 ms
WRT54GS (Broadcom BCM4712)	4154 ms	3816 ms	17323 ms	12739 ms

6 RELATED WORK

The anonymous submission of consumption data from a smart meter to another party for load reporting purposes has been previously studied by other authors. We here compare their solution with our solution.

Müller (Müller, 2010) gives a list of easily implementable recommendations to protect the customer's privacy.

1. *Reducing the Temporal Resolution*
   The fewer data points are submitted to the energy provider, the more difficult it is to identify the customer. Sharp peaks in the diagram are flattened and the energy consumption curve gets smoother, showing only average values.
2. *Aggregation of Intermediate Data*
   Data points of several days are accumulated at the same time and sent to the energy provider.
3. *Aggregation of Several Households*
   Like aggregation of intermediate data, data points can be aggregated over several households.

Reducing the temporal resolution or aggregating the data comes with the drawback of decreasing accuracy and response time. The first two recommendations contradict our no-delay-aim (aim 7). The aggregation of several households increases the freshness of the consumption data but comes with the drawback that a TTP is needed between smart meter and energy provider (contradicts aim 9). All smart meters send the consumption data to the TTP, which authenticates them, accumulates the data and forwards it to the energy provider. The third party must be independent from the energy provider to preserve the customer's privacy.

Petrlic (Petrlic, 2011) suggests a solution where a smart meter communicates with the energy provider over a collector. First, the smart meter encrypts the consumption data with the public key of the energy provider for load reporting. Second, it signs the encrypted message using a pseudonym and submits the signed and encrypted message to the collector. The collector verifies the signature, removes the signature and forwards the encrypted data signed by a MAC to the energy provider.
This solution requires a TTP between smart meter and energy provider. Another trusted party is needed to manage grid membership (contradicts aim 9).

Garcia and Jacobs suggest to use homomorphic encryption (Garcia and Jacobs, 2010), where each smart meter directly submits the consumption data to the energy provider without using a collector (they use a collector but this is part of the energy provider). The smart meter separates consumption data (every measurement) into N samples, where N is the number of smart meters in a neighbourhood. Every smart meter sends $N-1$ samples in an encrypted way to the neighbouring smart meters over the energy provider. All smart meters accumulate the consumption data (e.g. the first smart meter accumulates the first sample of all N consumption data, the second smart meter accumulates the second sample of all N consumption data...) and send the result back to the energy provider. A TTP is needed to create smart meter certificates (contradicts aim 9). Furthermore, a lot of encrypted messages have to be exchanged, especially if the neighbourhood is big. This fact makes the protocol inefficient.

7 CONCLUSIONS

The contribution of this paper consists in offering a new protocol for smart meters, which reports the electricity consumption back to their energy providers for the sake of load reporting and invoicing. Despite the fact that the customer's privacy is preserved, the protocol still allows the detection of unregistered smart meters and prevents spamming and replay attacks. Compared with other solutions, a TTP is not needed for key or group management, which makes our solution interesting for real world implementations. The security of the protocol is based on the well-investigated strong RSA assumption and the random oracle model. Benchmark results prove that the protocol is practical. The protocol is not restricted to electrical smart meters and can easily be adapted to

other sensor networks.

ACKNOWLEDGEMENTS

The author would like to thank Klaus Müller for contributing the data of figure 1.

REFERENCES

Berg Insight (2010). Worldwide installed base of smart electricity meters will reach 302.5 million units in 2015.

Camenisch, J. and Groth, J. (2005). Group Signatures: Better Efficiency and New Theoretical Aspects. In *proceedings of SCN '04, LNCS series*, pages 120–133. Springer.

Camenisch, J. and Lysyanskaya, A. (2003). A Signature Scheme with Efficient Protocols. In Cimato, S., Persiano, G., and Galdi, C., editors, *Security in Communication Networks*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer Berlin / Heidelberg. 10.1007/3-540-36413-7_20.

Camenisch, J. and Stadler, M. (1997). Proof Systems for General Statements about Discrete Logarithms. Technical Report 260, Institute for Theoretical Computer Science, ETH Zürich.

Dr. Neuhaus (2010). MUC-Controller zur Fernauslesung von Haushaltszählern via DSL – Specification.

Fiat, A. and Shamir, A. (1987). How to prove yourself: Practical solutions to identification and signature problems. In Odlyzko, A. M., editor, *Advances in Cryptology – CRYPTO '86*, volume 263, pages 186–194. Springer Verlag.

Garcia, F. D. and Jacobs, B. (2010). Privacy-friendly energy-metering via homomorphic encryption. In et al., J. C., editor, *In 6th Workshop on Security and Trust Management (STM 2010)*, Lecture Notes in Computer Science. Springer Verlag.

German Federal Ministry of Justice (2005). Gesetz über die Elektrizitts- und Gasversorgung (Energiewirtschaftsgesetz - EnWG), Paragraph 21b.

German Federal Ministry of Justice (2008). Erneuerbare-Energien-Gesetz, Paragraph 2.

Gladman, B. et al. (2010). MPIR - Multiple Precision Integers and Rationals.

Groth, J. (2005). Non-interactive zero-knowledge arguments for voting. In *In proceedings of ACNS '05, LNCS series*, pages 467–482. Springer-Verlag.

Lemay, M., Gross, G., Gunter, C. A., and Garg, S. (2007). Unified architecture for large-scale attested metering. In *in Hawaii International Conference on System Sciences. Big Island*. ACM.

McDaniel, P. and McLaughlin, S. (2009). Security and Privacy Challenges in the Smart Grid. *IEEE Security and Privacy*, 7(3):75–77.

Müller, K. J. (2010). Gewinnung von Verhaltensprofilen am intelligenten Stromzähler. *Datenschutz und Datensicherheit*, 6:359–364.

Müller, K. J. et al. (2011). Volkszähler - Smart Meter for Self-Construction.

Petrlic, R. (2011). A privacy-preserving concept for smart grids. In *18. DFN Workshop "Sicherheit in vernetzten Systemen"*.

Pointcheval, D. and Stern, J. (1996). Security proofs for signature schemes. In Maurer, U., editor, *Advances in Cryptology – EUROCRYPT '96*, volume 1070, pages 387–398. Springer Verlag.

Rabin, J. O. and Shallit, J. (1985). Randomized algorithms in number theory. Technical report, University of Chicago, Chicago, IL, USA.

Schnorr, C.-P. (1991). Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161 – 174.

The Tor Project (2011). Tor: anonymity online.

APPENDIX

This section contains a short summary of some cryptographic building blocks that we use in this paper. In the first part, we explain the Groth commitment scheme and give a short introduction to zero-knowledge protocols in the second part. At the end, we explain how to prove in zero-knowledge that a secret number fulfils an inequality.

A. Commitment Schemes

A non-interactive commitment scheme consists of three algorithms, *ComSetup*, *Commit* and *ComOpen*. The hiding property ensures that a commitment C_x to x does not reveal any information about x, whereas the binding property ensures that C_x cannot be opened to another value x'. For our solution, we choose the integer commitment scheme of Groth (Groth, 2005).

ComSetup: First generate the parameters of the commitment scheme. Set l_n as the bit length of a given special RSA modulus n and l as the bit-length of the security parameter. Randomly choose a generator $h \in_R QR_n$ and k exponents α_i (for $i = 1,\ldots,k$) of bit length $l_n + l$. Then compute $g_i = h^{\alpha_i}$ (for $i = 1,\ldots,k$).

Commit: To commit integers $(m_1,\ldots,m_k)$ of bit length l_m, choose a random integer p of bit length $l_n + l$ and compute $C = g_1^{m_1} \cdots g_k^{m_k} h^p \mod n$. Algorithm *Commit* outputs the commitment C together with auxiliary information p to open the commitment.

Open: To open a commitment C, compute $C' = g_1^{m'_1} \cdots g_k^{m'_k} h^{p'} \mod n$ for input integers $(m'_1,\ldots,m'_k)$ and p'. Check whether $C = C'$.

B. Zero-knowledge Proofs of Knowledge

With the help of a proof of knowledge a prover $\mathcal{P}$ can convince a verifier $\mathcal{V}$ that he knows a solution for a mathematical hard problem. On the one hand, an honest $\mathcal{P}$ can always convince $\mathcal{V}$ (*completeness* property); on the other hand, a dishonest $\mathcal{P}$ will fail to convince $\mathcal{V}$ with overwhelming probability if he attempts to cheat without knowing the correct solution (*soundness* property). A zero-knowledge proof of knowledge ZPK is a proof of knowledge where $\mathcal{V}$ obtains no further information from $\mathcal{P}$ other than the fact that $\mathcal{P}$ knows the solution of the underlying mathematically hard problem. E.g.:

$$\mathsf{ZPK}[(\omega) : x = g^{\omega}] \tag{2}$$

In ZPK (2) $\mathcal{P}$ proves to $\mathcal{V}$ the knowledge of the secret value w where w fulfils the relationship $x = g^w$. Secrets are marked in Greek characters. x and g are public values. Determining w by only knowing x and g is a mathematical hard problem.

ZPKs are usually implemented by Σ-protocols, which are based on the Schnorr protocol (Schnorr, 1991), an interactive, challenge-response protocol. To achieve a high level of security the protocol has to be executed several times, which decreases protocol performance. The Fiat-Shamir heuristic (Fiat and Shamir, 1987) (Pointcheval and Stern, 1996) and the random oracle model can be used to execute the protocol in a non-interactive way so that a ZPK becomes a signature proof of knowledge (SPK). Several predicates can be concatenated by logical AND/OR operators to model more complex proofs.

C. Inequality Proofs

Let $C_{w_i} = g^{w_i} h^{p_i}$ be Groth Commitments with integers w_i for $i = 1, \ldots, 3$. With the relation $g^{w_3} h^{p_3} = C_{w_1}^{w_2} h^{p_3} h^{-p_1 w_2}$ for $w_3 = w_1 \cdot w_2$ we can prove in zero-knowledge that w_3 is the product of w_1 and w_2:

$$\mathsf{ZPK}\left[\begin{array}{l}((\omega_i, \rho_i)_{i=1}^3, \omega_2 \cdot \rho_1) : \bigwedge_{i=1}^{3} C_{w_i} = g^{\omega_i} h^{\rho_i} \wedge \\ 1 = C_{\omega_1}^{\omega_2} g^{-\omega_3} h^{-\rho_1 \omega_2}\end{array}\right] \tag{3}$$

Lagrange's Four-Square Theorem says that every non-negative number w can be decomposed into a sum of four square numbers $w_1, \ldots, w_4$ with $w = (w_1)^2 + (w_2)^2 + (w_3)^2 + (w_4)^2$. If w has the form $4w' + 1$, Rabin and Shallit showed that even three squares $w_1^2, \ldots, w_3^2$ are enough to decompose w in probabilistic polynomial time (Rabin and Shallit, 1985). To prove that $4w' + 1 \geq 0$ where w' is a positive integer, we decompose $4w' + 1$ into a sum of three squares for a committed value w' and prove that $4w' + 1 = (w_1)^2 + (w_2)^2 + (w_3)^2$.

$$\mathsf{ZPK}\left[\begin{array}{l}(\omega', \rho, \omega_1, \omega_2, \omega_3) : C_w = g^{\omega'} h^{\rho} \wedge \\ 4\omega' + 1 = (\omega_1)^2 + (\omega_2)^2 + (\omega_3)^2\end{array}\right] \tag{4}$$

Proof (4) can be realized using the Σ^{GSP}-protocol and with the help of proof (3) (Groth, 2005).