Implementation of a Protocol for Secure Distributed Aggregation of Smart Metering Data

Cristina Rottondi*, Marco Savi*, Daniele Polenghi*, Giacomo Verticale*, and Christoph Krauß†
* Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza Leonardo da Vinci, 32, Milano, Italy
{rottondi,vertica}@elet.polimi.it,{daniele.polenghi,marco.savi}@mail.polimi.it
† Fraunhofer Research Institution for Applied and Integrated Security, Parkring 4, Garching b. Muenchen, Germany
christoph.krauss@aisec.fraunhofer.de

***Abstract*—Measurements gathered by Smart Meters and collected through the Automatic Metering Infrastructure of Smart Grids can be accessed by numerous external subjects for different purposes, ranging from billing to grid monitoring and management. Therefore, metering data must be securely handled, in order to protect the users' privacy and to prevent the disclosure of personal information through the analysis of energy consumption patterns.**

This paper proposes the implementation of a protocol for privacy-preserving aggregation of metering data in a distributed scenario, which relies on communication Gateways located in the customers' households. Measurements are encrypted by using a secret sharing scheme. The routing of the information flows is performed exploiting a variant of the Chord protocol. We evaluate the performance of the protocol and discuss how they are affected by two well known attacks to the Chord routing, namely the Sybil and Eclipse attacks.

***Index Terms*—Smart Grid; Multiparty Computation; Data Privacy; Chord; Sybil Attack; Eclipse Attack;**

I. Introduction

In the next years, the amount of user data collected by the Smart Grid is expected to dramatically increase with respect to the current electrical power grid: this arises great concerns regarding the privacy of the customers. The current electromechanical power meters installed at the customers' households will be replaced by "intelligent" digital devices called Smart Meters, which will provide to the Smart Grid not only information about the energy consumption, but also a great amount of user-related data which will be used by the utilities themselves (e.g., for billing purposes), by the grid managers (e.g., for electrical power state estimation) or by third parties (e.g., to provide value-added services, such as home energy consumption management). Since information about personal habits of the users can be deduced by analysing their energy consumption patterns, smart metering data should be collected in a privacy-preserving way, e.g., exploiting data aggregation, obfuscation or anonymization techniques [1].

In a previous paper [2], we have defined a framework for distributed aggregation of data gathered by Smart Meters and destined to multiple External Entities (EEs). The aggregation infrastructure relies on Gateways placed at the customers' premises, which process the metering data and encrypt them using a Hardware Security Module [3], so that only authorized parties have access to aggregated data. The routing of the information flows through the network is performed using a variant of the Chord protocol [4]. In this paper, we describe a prototype of the aggregation protocol in [2] and discuss how its performance are affected in case a collusion of malicious Gateways perform two of the most effective attacks to peer-to-peer Distributed Hash Table-based (DHT) networks, namely the *Sybil* [5] and the *Eclipse* [6] attacks.

The paper is structured as follows: Section II provides an overall view of the related work, while section III recalls the proposed aggregation architecture. The attacks to DHT routing is described in Section IV. Section V presents the implementation and the performance assessment of the protocol and Section VI presents the security discussion by analysing the impact of the *Sybil* and *Eclipse* attacks on the performance of the aggregation infrastructure. The paper is concluded in Section VII.

II. Related Work

Numerous aggregation schemes for smart metering data have been proposed in the recent literature: most of them rely on multiparty computation techniques, which allow the calculation of an aggregation function based on the inputs of various participants without disclosing the intermediate operations. In the context of Smart Grids, various aggregation protocols with different characteristics and complexities are proposed in papers [7]–[9], but none of them deals with a scenario where multiple external parties are interested in accessing the same data, aggregated according to different rules and granularities.

Another popular scenario in which distributed privacy-preserving data aggregation is applied is wireless sensor networks: in papers [10]–[12], different secure aggregation schemes and approaches to the routing of the data to be aggregated are discussed. However, most of the proposed solutions rely on the broadcasting nature of the wireless channels, while our proposal requires unicast channels and deals with geographically sparse nodes.

III. Review of the Aggregation Architecture

Fig. 1 shows the privacy-preserving aggregation architecture proposed in [2], which includes three sets of nodes: the *Meters*, M, which generate the energy consumption measurements, the *Gateways*, G, which collect and securely aggregate the metering data, and the *External Entities*, E, which are the

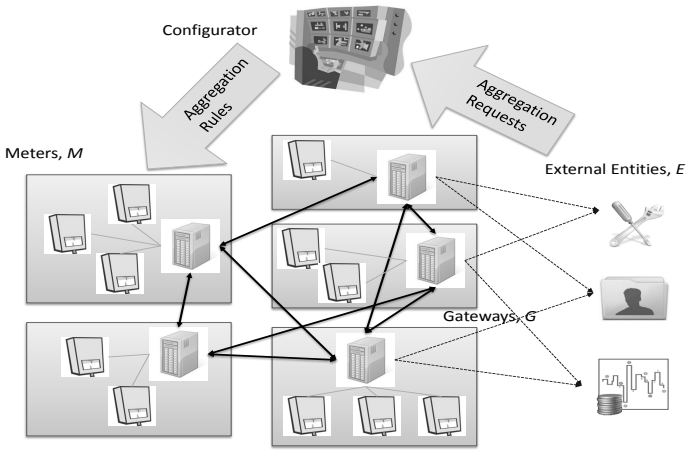

Fig. 1. The functional nodes of the architecture [2]

parties accessing the aggregated measurements. An additional node, the *Configurator*, collects the aggregation requests from the EEs (expressed in terms of sets of Meters they want to monitor), verify whether such requests are compliant to the security policies of the grid and allows or denies them accordingly.

Each Meter is directly connected to a Gateway, which receives data from a limited number of Meters (e.g., all the Meters in a building). At regular time intervals (e.g., every 15 mins), the Meter generates a measurement and sends it to the Gateway. The Gateway divides the measurements received from the local Meters in w shares using Shamir's Secret Sharing (SSS) scheme, a threshold cryptosystem which allows the reconstruction of the aggregated data in case $t \leq w$ shares are available, where t is a design parameter which defines the security level of the system. Moreover, the Gateways receive partially aggregated shares from other Gateways: since the SSS scheme has homomorphic properties with respect to addition, the s-th shares ($1 \leq s \leq w$) can be independently summed according to the aggregation rules specified by the EEs.

The aggregation is performed in a distributed fashion and the deployment of the information flows is performed using a variant of the Chord routing protocol, which creates w independent Chord rings, each responsible for the aggregation of one of the w shares. Every Gateway is placed in each of the w rings according to its Chord identifiers, obtained by hashing the node ID with a family of w independent hash functions. When a given EE obtains the approval of its aggregation request from the *Configurator*, w aggregation trees (one for each ring) are created, relying on the standard query routine of the Chord protocol. Then, at every time interval, data generated by the Meters are collected, divided in shares and aggregated by the Gateways belonging to the aggregation trees.

Once the aggregation process is completed, the aggregated shares are sent to the EEs. The EEs can recover the aggregated measurements through the Berlekamp-Welch algorithm [13], which allows a correct reconstruction in presence of l missing shares and e corrupted shares, provided that $w \geq t + 2e + l$.

IV. Attacks to P2P-based Aggregation

In this scenario, the EEs are assumed to behave according to the *honest-but-curious* attacker model, i.e., they cannot inject false messages or alter the routing of the communication flows, but try to deduce further information from the data they receive, possibly creating collusions. In contrast, in this paper the Gateways are assumed to behave as *dishonest* nodes, meaning that they can collude in order to alter the routing and the content of the messages. In particular, we focus on the *Sybil* and *Eclipse* attacks, which are representative of two classes of attacks to peer-to-peer distributed networks. In the former, an attacker runs multiple colluding Gateways in order to gain access to the measurements generated by a large number of Meters. In the latter, the colluding Gateways conspire to alter the construction of the aggregation trees by inducing the honest Gateways to select them as their neighbours, in order to mediate most of the aggregation requests specified by the EEs.

In both attacks, when a malicious Gateway receives an aggregation request for the measurements of a given Meter, it ignores the request and does not forward it to the Gateway locally connected to the Meter. Moreover, the malicious Gateways behave according to one of the following models:

- **Silent Gateway:** the Gateway does not provide any measurement, so that the final aggregated share is missing;
- **Liar Gateway:** the Gateway provides false or altered shares to its neighbours in the aggregation trees, so that the final aggregated share is corrupted and becomes unusable.

In addition to this, in the *Eclipse* attack the colluding Gateways modify their finger table[1] so that it only contains the identifiers of other colluding Gateways. This way, the probability of a malicious Gateway to be included in a generic aggregation tree is increased, since the finger tables are periodically exchanged and refreshed during the stabilization phase of the Chord protocol and whenever a new node joins/leaves the network.

Since we assume that all the communication channels are secure and authenticated, we do not consider the presence of external eavesdroppers.

V. Implementation and Performance Evaluation

The protocol has been implemented within the Twisted framework [14], a Python-based event-driven framework under the open source MIT license. Every node, with its own identifier, IP address and/or listening TCP port, generates a different Python process. The Chord routing has been implemented based on a local open source Python version of Chord, which computes the finger tables of every node in the network, given the identifiers of the nodes. The identifiers are computed by means of the MD5 hash function that takes as input the TCP listening port of the nodes and the Chord ring number s. We assume a static scenario i.e., the finger tables do not change during time, since there are no nodes joining or leaving the

[1]In the Chord protocol, the finger table is a local routing table which is used by each node to forward the queries

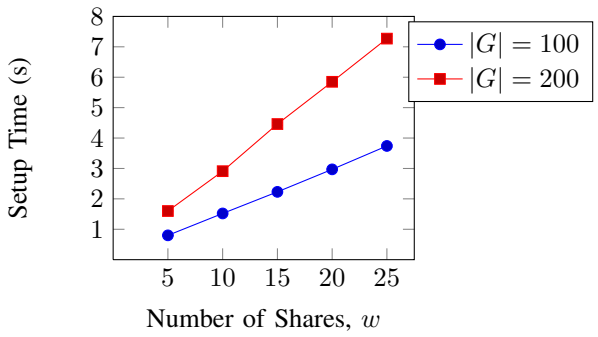

Fig. 2. Setup time required to create w aggregation trees, for different values of $|G|$.The precision of the 95%-confidence intervals (omitted in the plot) is below 10%.

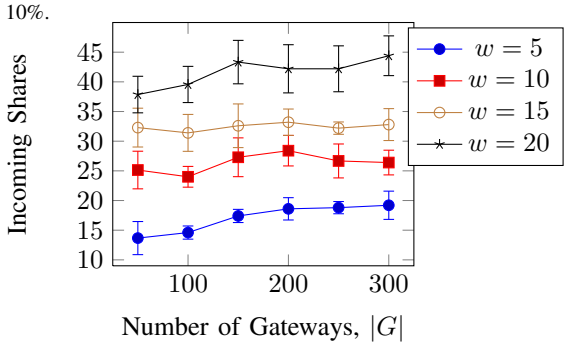

Fig. 3. Maximum number of incoming shares per Gateway, averaged over multiple instances (with 95%-confidence intervals), for different values of $|G|$ and w, assuming $|E| = 1$.

system. First, each node computes its finger table and creates the whole topology, i.e., the w different Chord rings (one for each share). Then, the nodes deploy the communication flows.

Note that all the connections are protected by SSL/TLS.The system comprises a Certification Authority (CA) which generates and certifies a public key/private key pair for every node with OpenSSL. Measurements have been performed using an Intel Xeon CPU model E5335 running at 2.00 GHz.

Figure 2 plots the time required to complete the setup phase of the protocol, i.e. to create the w aggregation trees to collect the aggregate measurements destined to each EE. The time is linear with respect to w and increases with the cardinality of G.

Figure 3 depicts the average number of shares that a Gateway is required to aggregate during the data collection phase of the protocol, for a single EE. Intuitively, the higher is the number of shares to be processed, the higher should be the computational capabilities of the Gateways. Results show the increasing trend of the computational effort at the Gateways with w and $|G|$.

VI. Security Evaluation

In this section, we evaluate the impact of the *Sybil* and *Eclipse* attacks on the performance of the aggregation protocol. For this purpose, the aggregation architecture and both attacks have been implemented within the *OMNET++/OverSim* framework [15], [16]. For the sake of simplicity, we assume that the underlying communication network is reliable and

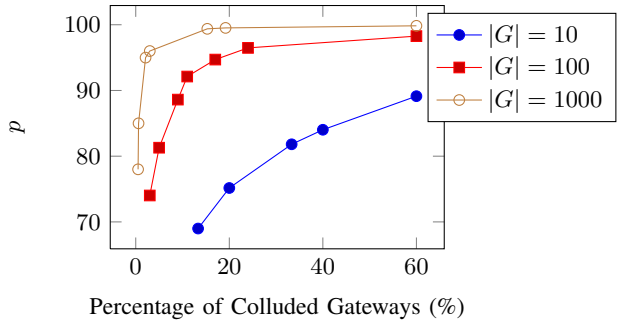

Fig. 4. Probability that the measurements generated by a given Meter are altered by one or more malicious Gateways, p, for the *Eclipse* attack.

timely, thus no shares can be lost due to communication errors or delays.

Let G_c be the number of colluded Gateways, let p be the probability that the measurements generated by a given Meter are altered by a malicious Gateway and let M_e be the cardinality of the set of Meters monitored by the EE e. The probability P_s that the s-th aggregated share is not corrupted is:

$$P_s = (1-p)^{M_e}$$

For the *Sybil* attack, simulation results show that p increases linearly with G_c.

Fig. 4 plots the trend of p as a function of the percentage of colluded Gateways, for the *Eclipse* attack. In this scenario, the malicious Gateways alter their finger tables by filling them only with the identifiers of other colluded nodes, which increases the probability that an aggregation requests is routed to a malicious Gateway. Therefore, p increases superlinearly with G_c: even with a small fraction of malicious Gateways, the probability p is very high and closely approaches 1 in case of large networks.

In both attacks, the probability P_r that the aggregated measurement requested by an EE is correctly recovered can be computed as follows. We first assume that the *Silent Gateway* model, in which some of the w aggregated shares will be missing at the EEs, but no shares are corrupted ($e = 0$). The aggregated measurements can be recovered through the Berlekamp-Welch algorithm if $l \leq w - t$.

Since the aggregate of M_e measurements can be retrieved in presence of up to $w - t$ missing shares, we obtain:

$$P_r = \sum_{i=0}^{w-t} \binom{w}{i} (1-P_s)^i P_s^{w-i} \quad (1)$$

Conversely, in the *Liar Gateway* model, the EEs will receive all the w shares ($l = 0$), but some of them will be corrupted. The Berlekamp-Welch algorithm allows the recovery of the aggregated measurements if the number of corrupted shares is bounded by $e \leq \frac{w-t}{2}$. Therefore, in this case we have:

$$P_r = \sum_{i=0}^{\lfloor \frac{w-t}{2} \rfloor} \binom{w}{i} (1-P_s)^i P_s^{w-i} \quad (2)$$

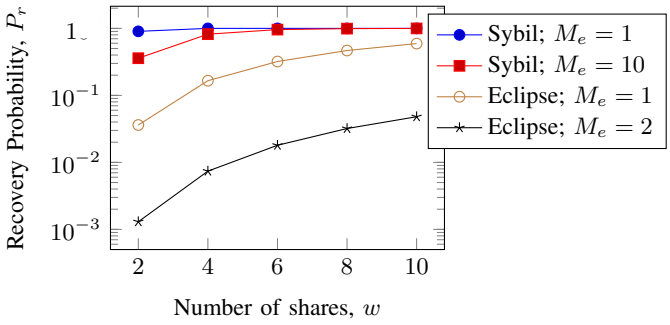

Fig. 5. Probability of correct recovery, P_r, assuming the *Silent* Gateway model, $|G| = 100$, $t = 2$, $G_c = 5$.

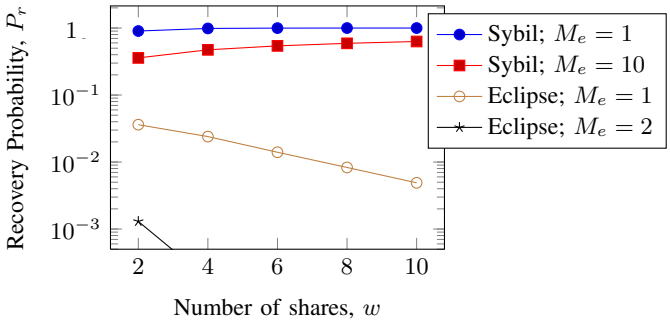

Fig. 6. Probability of correct recovery, P_r, assuming the *Liar* Gateway model, $|G| = 100$, $t = 2$, $G_c = 5$.

Fig. 5 plots the probability P_r that the aggregated measurement is correctly recovered by the EE, computed according to (1), as a function of the total number of shares w, for different values of M_e and assuming *Silent Gateway* model. While in the *Sybil* attack P_r is acceptable for small aggregates, the effect of the *Eclipse* attack is dire and makes the recovery of the aggregated measurements almost impossible even for very small values of M_e. However, increasing the total number of shares w improves the probability of recovery in all the considered cases. In Fig. 6, the same analysis is reported for the *Liar Gateway* model. With respect to the previous scenario, P_r is computed according to (2) and is always lower. Moreover, increasing w turns out not to be beneficial in the *Eclipse* attack, when P_s is extremely low.

Therefore, the usage of SSS scheme provides some countermeasures in case of missing shares, while the effect of injecting corrupted values leads to a strong degradation of the performance of the protocol. Furthermore, both attacks are more effective when the cardinality M_e of the set of monitored Meters is high.

VII. Conclusions

This paper describes the implementation of a protocol for secure collection of aggregated metering data. Measurements generated by Smart Meters are aggregated in a distributed fashion by exploiting the communication and cryptographic capabilities of Gateways located at the customers' premises. The routing of the information flows is deployed using a variant of the Chord protocol. We discuss the performance achieved by the implemented framework and how they are degraded in presence of the *Sybil* and *Eclipse* attacks modifying the Chord routing mechanisms and the message content. Results show that, under the assumption of a *Silent* model of malicious Gateway, in case of small aggregates the effects of both attacks can be partially compensated with a correct dimensioning of the number of shares in the SSS scheme. Conversely, in case of a *Liar* Gateway model, increasing the number of shares is beneficial only if the probability that the measurement generated by a Meter passes through one or more malicious Gateways is reasonably low (i.e., in the *Sybil* attack). In our opinion, future research should be focused on the techniques for the detection of colluding Gateways behaving according to the *Liar* model, which cause the most severe degradation in the performance of the aggregation protocol.

VIII. Acknowledgments

The authors would like to thank Stefano Mangioni for his useful contributions.

References

[1] National Institute of Standards and Technology (NIST), "Guidelines for smart grid cyber security," NIST Interagency Report 7628, Aug. 2010. [Online]. Available: http://www.nist.gov

[2] C. Rottondi, G. Verticale, and C.Krauß, "Distributed privacy-preserving aggregation of metering data in smart grids," (submitted).

[3] Federal Office for Information Security, "Protection profile for the gateway of a smart metering system," 2011. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/SmartMeter/PP-SmartMeter.pdf

[4] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," *Networking, IEEE/ACM Trans. on*, vol. 11, no. 1, Feb. 2003.

[5] J. Douceur, "The sybil attack," in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, P. Druschel, F. Kaashoek, and A. Rowstron, Eds. Springer Berlin / Heidelberg, 2002, vol. 2429, pp. 251–260.

[6] A. Singh, T. Ngan, P. Druschel, and D. Wallach, "Eclipse Attacks on Overlay Networks: Threats and Defenses," in *Proc IEEE INFOCOM*, Barcelona, Spain, April 2006.

[7] K. Kursawe, G. Danezis, and M. Kohlweiss, "Privacy-friendly aggregation for the smart-grid," in *Privacy Enhancing Technologies*, vol. 6794. Springer Berlin / Heidelberg, 2011, pp. 175–191.

[8] F. Li, B. Luo, and P. Liu, "Secure information aggregation for smart grids using homomorphic encryption," in *Smart Grid Communications (SmartGridComm) First IEEE Intl. Conf. on*, Oct. 2010, pp. 327–332.

[9] F. Garcia and B. Jacobs, "Privacy-friendly energy-metering via homomorphic encryption," in *6th Workshop on Security and Trust Management (STM 2010)*, 2010.

[10] H. Feng, G. Li, and G. Wang, "Efficient secure in-network data aggregation in wireless sensor networks," in *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, vol. 1, april 2010, pp. 194 –197.

[11] W. He, H. Nguyen, and X. L. et al., "iPDA: An integrity-protecting private data aggregation scheme for wireless sensor networks," in *Military Communications Conference, IEEE 2008*, nov. 2008, pp. 1 –7.

[12] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, may 2007, pp. 2045 –2053.

[13] N. Smart, *Cryptography: an Introduction*. McGraw-Hill, 2004.

[14] G. Lefkowitz. [Online]. Available: www.twistedmatrix.com

[15] A. Varga and R. Hornig, "An Overview of the OMNeT++ Simulation Environment," in *Simutools '08: Proceedings of the 1st International Conference on Simulation tools and techniques for Communications, Networks and Systems Workshops*, 2008.

[16] "OverSim: The Overlay Simulation Framework," http://www.oversim.org/.