WiP: Verifiable, Secure and Energy-Efficient Private Data Aggregation in Wireless Sensor Networks

Citation for the original published paper (version of record):
Tsaloli, G., Lancho Serrano, A., Mitrokotsa, K. et al (2022). WiP: Verifiable, Secure and Energy-Efficient Private Data Aggregation in Wireless Sensor Networks. Proceedings of ACM Symposium on Access Control Models and Technologies, SACMAT: 61-66. http://dx.doi.org/10.1145/3532105.3535040

WiP: Verifiable, Secure and Energy-Efficient Private Data Aggregation in Wireless Sensor Networks

Georgia Tsaloli
Chalmers University of Technology
Gothenburg, Sweden
tsaloli@chalmers.se

Alejandro Lancho
Massachusetts Institute of Technology
Cambridge, MA, USA
lancho@mit.edu

Katerina Mitrokotsa
University of St. Gallen
St. Gallen, Switzerland
katerina.mitrokotsa@unisg.ch

Giuseppe Durisi
Chalmers University of Technology
Gothenburg, Sweden
durisi@chalmers.se

ABSTRACT

Large amounts of data are collected by IoT devices, and transmitted wirelessly to cloud servers for aggregation. These data are often sensitive and need to remain secret. Moreover, the employed servers might be untrustworthy, and maliciously alter their results. To address this, public verifiability must be provided, *i.e.,* anyone can check the result's correctness. Nevertheless, any such protocol must also cope with the limited battery capacity of the IoT devices.

We investigate the problem of verifiable, privacy-preserving aggregation and how to accommodate the IoT energy-efficiency requirements, in the context of a 5G wireless communication setting. We revisit the verifiable additive homomorphic secret sharing (VAHSS), which computes the sum of K sensors' data. We propose a ***threshold*** secure and private VAHSS protocol where only a subset of the servers is needed for computing the sum. This sum is publicly verifiable thanks to a proof created during the protocol. We provide an energy efficiency analysis, including the trade-off between the sensors' transmitted power and the protocol's failure probability due to decoding errors in the wireless transmission phase.

CCS CONCEPTS

• **Security and privacy** → **Cryptography**; • **Networks** → *Network reliability.*

KEYWORDS

public verification, wireless communication, secret sharing, privacy

ACM Reference Format:
Georgia Tsaloli, Alejandro Lancho, Aikaterini Mitrokotsa, and Giuseppe Durisi. 2022. WiP: Verifiable, Secure and Energy-Efficient Private Data Aggregation in Wireless Sensor Networks. In *Proceedings of the 27th ACM Symposium on Access Control Models and Technologies (SACMAT '22), June 8–10, 2022, New York, NY, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3532105.3535040

SACMAT '22, June 8–10, 2022, New York, NY, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9357-7/22/06.
https://doi.org/10.1145/3532105.3535040

1 INTRODUCTION AND MOTIVATION

Future wireless communication networks are expected to support services and applications where a huge amount of battery-limited devices, sometimes referred to as IoT devices, connect wirelessly to the network to transmit data during very short periods of time. Due to their battery and power constraints, energy efficiency will play a crucial role in the design of these wireless networks.

This is the case, for instance, in smart metering applications, where sensors record data such as electricity consumption, and transmit such information to electricity suppliers or other organizations, which subsequently take actions based on the collected information. These sensors are battery-limited and assuming that they might be charged via a wired connection is unrealistic.

In this type of applications, preserving the privacy of the data collected by the sensors is needed to avoid undesirable inference of sensitive information. For example, suppose that an electricity supplier is interested in detecting energy losses to adjust the electricity production and provide better price packages than competitors. All this information can be acquired by establishing direct communication links between the sensors and the electricity supplier. By doing so, the electricity supplier would gain access also to sensitive information about the individuals, *e.g.,* how much time they spend at home or which devices they use, which may result in potential privacy issues. This implies that an electricity supplier should not be able to see each sensor's data in clear.

To summarize, in many applications as the one described above, resource constrained devices need to aggregate their collected data while, at the same time, keep the data private. To address this problem, we propose a privacy-preserving, cloud-assisted aggregation approach, where multiple, possibly untrusted servers, perform joint additions based on the inputs received from each of the sensors (IoT devices). The use of multiple servers aims to avoid placing all the trust into a single server, which might fail. Privacy and security in the entire process is guaranteed by our proposed ***threshold*** verifiable additive homomorphic secret sharing (VAHSS) construction. The term ***threshold*** refers to the ability of our proposed protocol to work using only a subset of the available servers rather than requiring all of them for the computations. This solution modifies the VAHSS protocol, introduced by Tsaloli *et al.* [12], which is based on homomorphic hash functions. Specifically, modifying certain algorithms of their solution, our new protocol can perform a successful computation as long as a number of servers above a predefined threshold, rather than all of them, participates in the computation. In addition, by employing the proposed threshold VAHSS protocol, we provide public verifiability guarantees: anyone is able to verify that the result of the cloud-assisted aggregation is correct. Finally, we analyze the power level at which the IoT sensors need to transmit for the aggregation process to be executed successfully over a wireless sensor network. Determining such power level is crucial, since the IoT devices need to be operated in an energy efficient way, to preserve their battery. The specific contributions of the paper are summarized below:

- We propose a ***threshold*** VAHSS protocol that handles failure or dropouts of servers. The standard VAHSS requires the participation of all servers. Thus, it is not robust against servers' failures. On the contrary, our solution requires only a subset of untrusted servers to provide the result of the privacy-preserving aggregation and assure its correctness.
- We provide an analysis of the performance of the VAHSS protocol in a wireless network scenario, where the sensors are served by a 5G base station (BS) equipped with a large number of antenna arrays. Using the framework recently introduced in [10], we analyze the so called *network availability*, as a function of the transmit power used by each sensor. The network availability is defined as the fraction of sensor placements within a given area for which the average probability that the VAHSS protocol fails is below a given target, for a given power budget and latency constraints. Using this performance metric, we characterize the energy efficiency as a function of the threshold of the protocol.

Organization. Section 2 summarizes some background material. In Section 3, we present the proposed ***threshold*** VAHSS protocol and discuss how we may deal with malicious users' inputs. In Section 4, we describe the communication setup and provide the theoretical framework used to assess the performance of the protocol in a 5G wireless sensor network. In Section 5, we provide some numerical results. Section 6 reports the related work. Finally, we draw some conclusions in Section 7.

2 PRELIMINARIES

According to the definition of verifiable homomorphic secret sharing (VHSS) proposed in [11], a K-user, S-server, t-secure VHSS scheme for a function $f : \mathcal{X} \mapsto \mathcal{Y}$, is a seven-tuple of probabilistic polynomial time (PPT) algorithms (**Setup**, **ShareSecret**, **PartialEval**, **PartialProof**, **FinalEval**, **FinalProof**, **Verify**), which satisfy *correctness*, *verifiability*, and *security* as defined below.

- **Correctness**: for any secret input $x_1, \ldots, x_K$, for all S-tuples in the set $\{(\mathsf{share}_{i1}, \ldots, \mathsf{share}_{iS}), \tau_i\}_{i=1}^{K}$ coming from the algorithm **ShareSecret**, for all $y_1, \ldots, y_S$ computed by the algorithm **PartialEval**, $\sigma_1, \ldots, \sigma_{|\kappa|}$[1] computed by the algorithm **PartialProof**, and for y and σ generated by **FinalEval** and **FinalProof**, respectively, the scheme should satisfy the following correctness requirement:
$$\Pr\left[\ \mathbf{Verify}(pp, \sigma, y) = 1\ \right] = 1,$$
where pp denotes any public parameters needed.
- **Verifiability**: let T be the set of corrupted servers with $|T| \leqslant S$ (note that, for $|T| = S$, the verifiabililty property holds; however, we do not have a secure system). Denote, by $\mathcal{A}$, any PPT adversary and consider K secret inputs $x_1, \ldots, x_K \in \mathbb{F}$. Any PPT adversary $\mathcal{A}$ who controls the shares of the secret inputs for any j, such that $\mathsf{server}_j \in T$ can cause a wrong value to be accepted as $f(x_1, x_2, \ldots, x_K)$ with negligible probability. We define the experiment $\mathbf{Exp}^{\text{Verif.}}_{\text{VHSS}}(x_1, \ldots, x_K, T, \mathcal{A})$:
  1. For all $i \in [K]$, generate $(\mathsf{share}_{i1}, \ldots, \mathsf{share}_{iS}, \tau_i) \leftarrow$ **ShareSecret**$(1^\lambda, i, \boldsymbol{x}_i)$ and publish τ_i.
  2. For all j, such that $\mathsf{server}_j \in T$, give to the adversary the shares $(\mathsf{share}_{1j}, \ldots, \mathsf{share}_{Kj})^\intercal$.
  3. For each corrupted $\mathsf{server}_j \in T$, the adversary $\mathcal{A}$ outputs modified shares $y_j{}'$ and $\sigma_k{}'$. Subsequently, for j, such that $\mathsf{server}_j \notin T$, set $y_j{}' =$ **PartialEval**$(j, (x_{1j}, \ldots, x_{Kj}))$ and $\sigma_k{}' =$ **PartialProof**$(sk, pp, \text{secret}_{\text{values}}, k)$. Note that we consider modified $\sigma_k{}'$ only when computed by the servers.
  4. Compute $y' =$ **FinalEval**$(y_1{}', \ldots, y_S{}')$, *i.e.*, the modified final result, and the modified final proof $\sigma' =$ **FinalProof**$(pp, \sigma'_1, \ldots, \sigma'_{|k|})$.
  5. If $y' \neq f(x_1, x_2, \ldots, x_K)$ and **Verify**$(pp, \sigma', y') = 1$, then output 1 else 0.

  We require that, for any K secret inputs $x_1, \ldots, x_K \in \mathbb{F}$, any set T of corrupted servers and any PPT adversary $\mathcal{A}$ it holds:
$$\Pr[\mathbf{Exp}^{\text{Verif.}}_{\text{VHSS}}(x_1, x_2, \ldots, x_K, T, \mathcal{A}) = 1] \leq \varepsilon, \text{for some negligible } \varepsilon.$$
- **Security**: let T be the set of the corrupted servers with $|T| < S$. Consider the following semantic security challenge experiment:
  1. The adversary $\mathcal{A}_1$ gives $(i, x_i, x'_i) \leftarrow \mathcal{A}_1(1^\lambda)$ to the challenger, where $i \in [K]$, $x_i \neq x'_i$ and $|x_i| = |x'_i|$.
  2. The challenger picks a bit $b \in \{0, 1\}$ uniformly at random and computes $(\widehat{\mathsf{share}}_{i1}, \widehat{\mathsf{share}}_{i2}, \ldots, \widehat{\mathsf{share}}_{iS}, \widehat{\tau}_i) \leftarrow$ **ShareSecret**$(1^\lambda, i, \hat{\boldsymbol{x}}_i)$ where the secret input is $\hat{\boldsymbol{x}}_i = \begin{cases} x_i, \text{if } b = 0 \\ x'_i, \text{otherwise} \end{cases}$.
  3. Given the shares from the corrupted servers T and $\widehat{\tau}_i$, the adversary distinguisher outputs a guess $b' \leftarrow \mathcal{D}((\widehat{\mathsf{share}}_{ij})_{j|\mathsf{server}_j \in T}, \widehat{\tau}_i)$.

  Let $\mathrm{Adv}(1^\lambda, \mathcal{A}, T) := \Pr[b = b'] - 1/2$ be the advantage of $\mathcal{A} = \{\mathcal{A}_1, \mathcal{D}\}$ in guessing b in the above experiment, where the probability is taken over the randomness of the challenger and of $\mathcal{A}$. A VHSS scheme is t-secure if, for all $T \subset \{\mathsf{server}_1, \ldots, \mathsf{server}_S\}$ with $|T| \leq t$, and all PPT adversaries $\mathcal{A}$, it holds that $\mathrm{Adv}(1^\lambda, \mathcal{A}, T) \leq \varepsilon(\lambda)$ for some negligible $\varepsilon(\lambda)$.

3 THRESHOLD VERIFIABLE ADDITIVE HOMOMORPHIC SECRET SHARING

In this section, we propose a method that achieves privacy-preserving aggregation when multiple users outsource their inputs to multiple untrusted servers. The method is based on the work proposed by Tsaloli *et al.* [11], which provides VAHSS using homomorphic hash functions. With our proposed threshold VAHSS, we

[1] $\kappa = i$ if partial proofs are generated by the users or, otherwise, $\kappa = j$ if they are generated by the servers. Thus, $|\kappa| = K$ or $|\kappa| = S$, respectively.

provide stronger fault tolerance and resilience guarantees, since only a subset (threshold) of the untrusted servers need to participate in the aggregation process. More precisely, denoting the threshold by t, our proposed protocol requires $t + 1$ out of S servers to successfully perform the computation. This means that, even if some of the servers drop or fail to collect all necessary values, our proposed solution for privacy-preserving aggregation still works and provides the sum of the collected data and a proof of correctness of the sum. In addition, we address not only the possible malicious behavior from the servers (publicly verifiable aggregation) but we also investigate how we may avoid possible malicious behavior from the users (e.g., possibly injection of wrong inputs to disrupt the aggregation computation). Precisely, we suggest the employment of range proofs from the user side to avoid their potentially malicious behavior and limit the range of acceptable inputs.

In the proposed threshold VAHSS protocol, K users (*e.g.,* sensors[2]) outsource the aggregation of their joint inputs to S untrusted servers. Each sensor holds an input value x_i that must remain secret. Values related to these secret inputs—the so-called shares of the secret inputs—are distributed to the S servers. Note that the shares of the secrets reveal neither the secret itself nor sensitive information about it. Each server is expected to perform computations on those shares to provide some results (partial evaluations) needed for the protocol. The goal is to compute the sum of the secret inputs, namely to compute $f(x_1, \dots, x_K) = x_1 + \dots + x_K$. This becomes possible by utilizing values computed throughout the protocol by the servers, without knowing the secret inputs but rather knowing shares of them and values that come from suitable combinations of the shares. Apart from the sum value of the sensors' secrets, denoted by y, the protocol provides a proof σ of the correctness of y. An overview of the protocol setting is illustrated in Fig. 1.

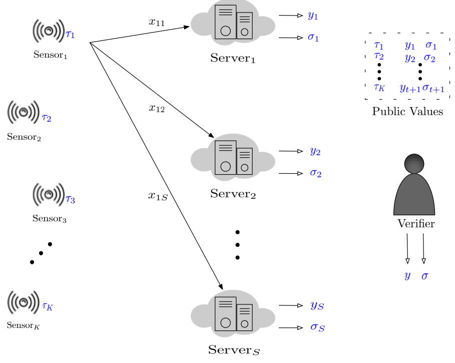

Figure 1: Overview of the *threshold* VAHSS setting

3.1 Threshold VAHSS Construction

In this section, we present the proposed ***threshold*** VAHSS construction. Considering the realistic scenario of servers failing (dropping out) during the execution of the protocol, we suggest a solution that addresses this issue and we show that it is correct, secure and verifiable. We must highlight that even though [11] provides a VAHSS construction using homomorphic hash functions, their solution requires all the S partial values for computing both the sum y and the corresponding proof of correctness. Hence, it is not robust against servers' failures. The algorithms of the ***threshold*** VAHSS construction are described below:

- **ShareSecret**. This algorithm takes as input the secret x_i that corresponds to the i-th sensor. Let $\mathbb{F} = \mathbb{F}_N$. Then, for randomly selected values $\{a_i\}_{i \in \{1,\dots,t\}} \in \mathbb{F}$, a polynomial $p_i(X) = x_i + a_1X + a_2X^2 + \dots + a_tX^t$ of degree t is formed. For distinct non-zero field elements $\theta_{i1}, \dots, \theta_{iS}$ and the "Lagrange coefficients" $\lambda_{i1}, \dots, \lambda_{iS}$, it holds that for any univariate polynomial p_i of the presented form we have $p_i(0) = \sum_{j=1}^{t+1} \lambda_{ij} p_i(\theta_{ij})$. Note that the index i stands for each sensor$_i$. By definition, for any $i \in [K], \lambda_{1j} = \dots = \lambda_{nj} = \Lambda_j$ and for a given $j \in [S], \theta_{1j} = \dots = \theta_{Kj} = \Theta_j$. Define $x_{ij} = p_i(\Theta_j)$ to be the share of the secret x_i corresponding to the server$_j$. For a pseudorandom function (PRF) $F : \{0,1\}^{l_1} \times \{0,1\}^{l_2} \mapsto \mathbb{F}$, define $R_i = F_k(i, file_i)$ for a sensor's key $k \in \{0,1\}^{l_1}$ and an input $file_i$ associated with sensor$_i$ such that $(i, file_i) \in \{0,1\}^{l_2}$. For $i = K$ we require $\mathbb{F} \ni R_K = \phi(N)\lceil \frac{\sum_{i=1}^{K-1} R_i}{\phi(N)} \rceil - \sum_{i=1}^{K-1} R_i$. Next, consider the collision-resistant homomorphic hash function [13] $H : x \mapsto g^x$ (with g a generator of the multiplicative group of $\mathbb{F}$) proposed by Krohn *et al.* [8]. Lastly, this algorithm computes a value τ_i defined as $\tau_i = H(x_i + R_i)$. The output of **ShareSecret** is $((\Theta_1, x_{i1}), (\Theta_2, x_{i2}), \dots, (\Theta_S, x_{iS}), \tau_i)$.
- **PartialEval**. For a given j as well as the K shares $(x_{1j}, x_{2j}, \dots, x_{Kj})$ that correspond to it, the algorithm sums the shares and produces $x_{1j} + \dots + x_{Kj} = p_1(\Theta_j) + \dots + p_K(\Theta_j) = \sum_{i=1}^{K} p_i(\Theta_j) = y_j$. The output of **PartialEval** is y_j.
- **PartialProof**. This algorithm takes as input the j-th shares $(x_{1j}, x_{2j}, \dots, x_{Kj})$ of the K secrets and, for the hash function previously described, computes $g^{\sum_{i=1}^{K} x_{ij}} = g^{y_j} = H(y_j) = \sigma_j$. The output of **PartialProof** is σ_j.
- **FinalEval**. For the selected ***threshold*** set of $t + 1$ servers (*w.l.o.g.* we consider the first $t + 1$ of them), given the partial computed values $y_1, \dots, y_{t+1}$, this algorithm computes $\Lambda_1 y_1 + \dots + \Lambda_j y_{t+1} = \sum_{j=1}^{t+1} \Lambda_j y_j = y$. The output of **FinalEval** is y.
- **FinalProof**. This algorithm requires ***threshold*** $t+1$ amount of partial proofs $\sigma_1, \dots, \sigma_S$ to generate $\prod_{j=1}^{t+1} {\sigma_j}^{\Lambda_j} = \sigma$. The output of **FinalProof** is σ.
- **Verify**. Given the public values $\tau_1, \dots, \tau_K$, a proof σ and a value y, the algorithm performs the following boolean check for the described hash function H: $\sigma = \prod_{i=1}^{K} \tau_i \wedge \prod_{i=1}^{K} \tau_i = H(y)$. The output of **Verify** is 1 if the check passes and, thus, y is indeed the sum value, or 0 otherwise.

The algorithms of the protocol are executed by the different participating parties to produce the sum value y and a proof σ that certifies its correctness. More precisely, each sensor$_i$ holds a secret input x_i and executes **ShareSecret** to generate the necessary shares that are distributed to each of the S servers. Each sensor publishes also a value τ_i that is needed later on for the verification. Next, each server$_j$ executes both **PartialEval** and **PartialProof** to compute and publish a partial value y_j and a partial proof σ_j, respectively. Note here that these values are generated using shares of the secret

[2] In the rest of the paper, the words sensors and users are used interchangeably.

inputs from the sensors rather than the actual secret inputs. In other words, the servers do not have access to the secret data of sensors but only to shares of them that do not reveal sensitive information about x_i's. Then, **FinalEval** requires a ***threshold*** of $t+1$ amount of servers' values to form the final sum y. Respectively, **FinalProof** uses the same amount of servers' partial proofs to generate the final proof of correctness σ that is to be used for the verification. The algorithms **FinalEval**, **FinalProof** and **Verify** are executed by any verifier that is interested in the result of the computation. More precisely, the verifier uses y, σ and the τ_i's to execute **Verify** and check if those values match. If the check succeeds the verifier gets the sum y, otherwise it rejects the result of the computation. Note that, with respect to the work proposed by Tsaloli *et al.* [11], to achieve a ***threshold*** construction, we have modified i) the shares that every user generates in **ShareSecret**, ii) how the **FinalEval** algorithm reconstructs the computed sum as well as iii) how **FinalProof** generates the final proof that is used for the verification. Our solution is correct, t-secure and verifiable. Being correct means that it provides the sum value y such that $y = x_1 + \ldots + x_K$ as expected. It is verifiable and, therefore, a wrong value y' fails to pass the protocol. Lastly, the t-security of the protocol means that t malicious adversaries which collude are not able to break the protocol and get the secret inputs of the sensors. Recall that $t+1$ servers are required for the aggregation. Thus, only one of them needs to be honest.

Correctness. To prove correctness, we need to show that: $\Pr\left[\ \mathbf{Verify}(\tau_1, \ldots, \tau_K, \sigma, y) = 1\ \right] = 1$. By construction, we get:

$$\begin{aligned} y &= \sum_{j=1}^{t+1} \Lambda_j y_j = \sum_{j=1}^{t+1} \Lambda_j \sum_{i=1}^{K} p_i(\Theta_j) = \sum_{i=1}^{K} \sum_{j=1}^{t+1} \Lambda_j \cdot p_i(\theta_j) \\ &= \sum_{i=1}^{K} \sum_{j=1}^{t+1} \lambda_{ij} \cdot p_i(\theta_{ij}) = \sum_{i=1}^{K} p_i(0) = \sum_{i=1}^{K} x_i \end{aligned} \tag{1}$$

Next, for the final proof σ, by construction, it holds that:

$$\begin{aligned} \sigma &= \prod_{j=1}^{t+1} \sigma_j^{\Lambda_j} = \prod_{j=1}^{t+1} H(y_j)^{\Lambda_j} = \prod_{j=1}^{t+1} (g^{y_j})^{\Lambda_j} = \prod_{j=1}^{t+1} g^{\Lambda_j y_j} \\ &= g^{\sum_{j=1}^{t+1} \Lambda_j y_j} \overset{see\ Eq.\ (1)}{=} g^y = H(y) \end{aligned}$$

$$\begin{aligned} \text{and} \quad \prod_{i=1}^{K} \tau_i &= \prod_{i=1}^{K} g^{x_i + R_i} = g^{\sum_{i=1}^{K} x_i} g^{\sum_{i=1}^{K} R_i} \\ &= g^{\sum_{i=1}^{K} x_i} g^{\sum_{i=1}^{K-1} R_i + R_K} = g^{\sum_{i=1}^{K} x_i} g^{\phi(N) \lceil \frac{\sum_{i=1}^{K-1} R_i}{\phi(N)} \rceil} \\ &= g^{\sum_{i=1}^{K} x_i} = g^{x_1 + \ldots + x_K} \overset{see\ Eq.\ (1)}{=} g^y = H(y) \end{aligned}$$

Therefore, we get that $\sigma = \prod_{i=1}^{K} \tau_i \wedge \prod_{i=1}^{K} \tau_i = H(y)$ holds. Thus, **Verify** outputs 1 with probability 1, as expected. □

The t-security and the verifiability proofs follow from [11].

VAHSS against malicious users. Another aspect that can be incorporated in this protocol is that we can prevent malicious users from disrupting the computed result with inputs that are not allowed. In fact, it is possible to incorporate a proof of range [5] on the users' inputs in the VAHSS protocol to protect the computation from malicious users. More precisely, each sensor$_i$ (user) that holds a secret input x_i can be expected to prove that its secret value lies in the range $[0, 2^l - 1]$, where l stands for the bit length of the range. Let $\mathbb{G}$ be a cyclic group of prime order p, $\mathbb{Z}_p$ denoting the ring of integers modulo p, V be a Pedersen commitment and g, h be generators of $\mathbb{G}$. Then, the range proof system proves the following relation:

$$\{(g, h \in \mathbb{G}, V, l;\ x_i, \rho \in \mathbb{Z}_p) : V = h^{\rho} g^{x_i} \wedge x_i \in [0, 2^l - 1]\}. \tag{2}$$

Following the work of Bünz *et al.* [5], the relation (2) is transformed to a single inner-product identity check. To generate this proof, the prover, *i.e.,* sensor$_i$ uses a non-interactive protocol that is secure and full zero-knowledge in the random oracle model, using Fiat-Shamir heuristic [3]. This protocol proves the required relation that assures that x_i is within the range, without revealing the secret input x_i. The prover, *i.e.,* sensor$_i$ sends only $2 \cdot \lceil \log_2 l \rceil + 4$ group elements and 5 elements in $\mathbb{Z}_p$. Therefore, sensor$_i$ may efficiently and non-interactively provide a proof such that a verifier can confirm that the input is within an acceptable range without seeing x_i itself.

4 COMMUNICATION-THEORETICAL FRAMEWORK

We consider the scenario depicted in Fig. 2 where K sensors are served by a 5G BS equipped with $M \gg 1$ antennas. Such a setup is commonly referred to as massive multiple-input multiple-output (MIMO), and it is one of the main technological components of 5G. Once a share transmitted by a sensor is received by the BS, it is forwarded to the relevant server. We assume that the BS is a trusted entity that always forwards the received shares to the correct server. We also assume that the links between the BS and the servers are error-free optical links and focus exclusively on the decoding errors occurring in the wireless interface between sensors and BS.

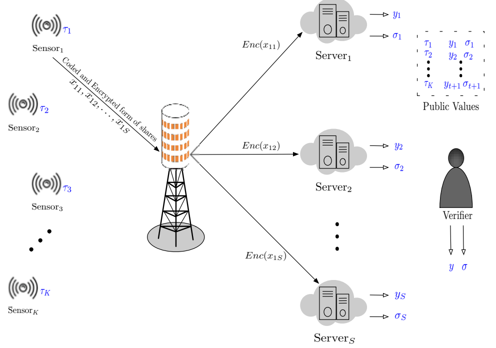

Figure 2: Integration of the *threshold* VAHSS into a 5G wireless sensor network supported by a Massive MIMO BS.

The sensors transmit their shares synchronously, starting from the ones intended for server$_1$, in different transmission rounds, for a total of S rounds, when they transmit the shares intended for server$_S$. In each transmission round, the received signal at the BS contains the superposition of the shares sent from all the sensors. Thanks to the large antenna array, the BS is able to separate the information streams pertaining to different users by performing simple linear processing operations [4].

We assume that each share is encrypted, to guarantee security in the wireless transmission phase. Indeed, the wireless channel is accessible to any eavesdropper. Furthermore, each encrypted share is embedded into a coded packet, to protect the share against errors occurring in the wireless transmission phase, due *e.g.*, to noise and residual interference after linear processing.

To assess the performance of the threshold VAHSS protocol in this scenario, we use the framework introduced in [10], which provides a rigorous characterization of the coded-packet error probability, as a function of system parameters such as, *e.g.*, transmit power, transmission rate, number of BS antennas, and number of sensors. Let this coded-packet error probability be ϵ, which we shall characterize shortly. Under the assumption that packet error events are independent across transmission rounds, the probability $\mathrm{P_e}$ that the threshold VAHSS protocol fails because of missing shares can be upper-bounded as $\mathrm{P_e} \le \sum_{r=S-t}^{S} \binom{S}{r}(K\epsilon)^r$ (3). In words, the term $K\epsilon$ is an upper bound on the probability that at least one share is decoded incorrectly in a given transmission round, resulting in a server to be excluded from the cloud computation. Hence, $\binom{S}{r}(K\epsilon)^r$ is an upper bound on the probability that r servers are excluded from the cloud computation.

To estimate ϵ, we present a mathematical model of the communication links between the BS and a given sensor. We consider a discrete-time complex-baseband equivalent of the channel input-output relation and assume that each coded packet spans n discrete-time channel uses. For a given arbitrary transmission round, the received signal $\mathbf{r}[k] \in \mathbb{C}^M$ at the BS at discrete-time $k \in [n]$ is modeled as $\mathbf{r}[k]=\sum_{i=1}^{K} \mathbf{h}_i q_i[k]+\mathbf{z}[k]$. Here, $\mathbf{h}_i \in \mathbb{C}^M$ denotes the channel vector between sensor$_i$ and the BS. We use a spatially-correlated Rayleigh-fading model [4], where $\mathbf{h}_i \sim \mathcal{CN}(\mathbf{0}_M, \mathbf{R}_i)$ remains constant for the duration of a packet transmission, but changes independently across transmission rounds. $\mathbf{R}_i$ describes the spatial channel correlation between sensor$_i$ and the BS antennas, which we assume to be known at the BS [4, Sec. 2.2]. The vector $\mathbf{z}[k] \in \mathbb{C}^M$, with independent and identically distributed (i.i.d.) elements distributed as $\mathcal{CN}(0, \sigma_{\mathrm{bs}}^2)$, models the additive noise. Finally, $q_i[k]$ is the k symbol of the coded packet transmitted by sensor$_i$.

We assume that the first $n_\mathrm{p} \ge K$ symbols in each coded packet are used to transmit pilot symbols used by the BS to estimate the channels, whereas the remaining $n_\mathrm{d} = n - n_\mathrm{p}$ symbols contain the data. The n_p-length pilot sequence of sensor$_i$ is denoted by the vector $\boldsymbol{\phi}_i \in \mathbb{C}^{n_\mathrm{p}}$. It is designed so that $\|\boldsymbol{\phi}_i\|^2 = n_\mathrm{p}$. We also assume that the sequences used by the K sensors are mutually orthogonal. We use minimum mean-square error (MMSE) channel estimation [4, Sec. 3.2], for which the estimate $\widehat{\mathbf{h}}_i$ of $\mathbf{h}_i$ is given by

$$\widehat{\mathbf{h}}_i = \sqrt{\rho n_\mathrm{p}} \mathbf{R}_i \mathbf{Q}_i \left(\mathbf{V}^{\mathrm{pilot}} \boldsymbol{\phi}_i\right), \text{where}$$

$$\mathbf{Q}_i = \left(\sum_{i'=1}^{K} \rho \mathbf{R}_{i'} \boldsymbol{\phi}_{i'}^{\mathrm{H}} \boldsymbol{\phi}_i + \sigma^2 \mathbf{I}_M\right)^{-1}, \mathbf{V}^{\mathrm{pilot}} = \sum_{i=1}^{K} \sqrt{\rho} \mathbf{h}_i \boldsymbol{\phi}_i^{\mathrm{T}} + \mathbf{Z}^{\mathrm{pilot}}.$$

Here, $\mathbf{Z}^{\mathrm{pilot}} \in \mathbb{C}^{M \times n_\mathrm{p}}$ is the additive noise with i.i.d. elements distributed as $\mathcal{CN}(0, \sigma_{\mathrm{bs}}^2)$.

We assume that the BS uses the channel estimates $\{\widehat{\mathbf{h}}_i\}_{i=1}^{M}$ to separate the users via linear combining. Specifically, to recover the signal transmitted by sensor$_i$, it projects the vector $\mathbf{r}[k]$, $k = n_\mathrm{p} + 1, \ldots, n$, onto the MMSE linear combiner $\mathbf{u}_i$ given by $\mathbf{u}_i = \left(\sum_{i'=1}^{K} \widehat{\mathbf{h}}_{i'} \left(\widehat{\mathbf{h}}_{i'}\right)^{\mathrm{H}} + \mathbf{Z}\right)^{-1} \widehat{\mathbf{h}}_i$, with $\mathbf{Z} = \sum_{i'=1}^{K} \mathbf{\Phi}_{i'} + \frac{\sigma_{\mathrm{bs}}^2}{\rho} \mathbf{I}_M$, where $\mathbf{\Phi}_i = \rho n_\mathrm{p} \mathbf{R}_i \mathbf{Q}_i \mathbf{R}_i$. Let $v[k] = \mathbf{u}_i^{\mathrm{H}} \mathbf{r}[k]$ and $g = \mathbf{u}_i^{\mathrm{H}} \mathbf{h}_i$. Then, we can express $v[k]$ as $v[k] = g q_i[k] + z[k]$, where $z[k] = \sum_{i'=1, i' \neq i}^{K} \mathbf{u}_i^{\mathrm{H}} \mathbf{h}_{i'} q_{i'}[k] + \mathbf{u}_i^{\mathrm{H}} \mathbf{z}[k]$ contains the additive noise and the residual interference from the other sensors after combining.

To obtain an estimate of the packet error probability, we assume that the BS treats the channel estimate as perfect and the residual interference as noise by performing scaled nearest-neighbor decoding. A random coding analysis, performed under the assumptions that the data symbols $q_i[k]$ are i.i.d. and follow a $\mathcal{CN}(0, \rho)$ distribution, reveals that ϵ can be upper-bounded as [10, Eq. (3)]

$$\epsilon \le \Pr\left[\sum_{k=n_\mathrm{p}+1}^{n} \imath_s(q[k], v[k]) \le \log\left(\frac{2^b - 1}{u}\right)\right]. \tag{4}$$

Here b is the number of bits needed to describe the encrypted share, u is a uniformly distributed random variable on the interval $[0, 1]$, and

$$\imath_s(q[k], v[k]) = -s\,|v[k] - \widehat{g} q[k]|^2 + \frac{s\,|v[k]|^2}{1 + s\rho\,|\widehat{g}|^2} + \log\left(1 + s\rho\,|\widehat{g}|^2\right)$$

with $\widehat{g} = \mathbf{u}_i^{\mathrm{H}} \widehat{\mathbf{h}}_i$, and $s > 0$ being an optimization parameter that can be used to tighten the bound.

5 NUMERICAL RESULTS

The simulation setup is based on the one used in [10] to assess the performance of ultra-reliable short-packet communications with massive MIMO. Specifically, the simulation setup consists of a square area of size 500 m × 500 m containing $K = 50$ sensors independently and uniformly distributed within the area at a distance of at least 5 m from the BS. The BS is placed at the center of the considered area. We consider a horizontal uniform linear array with $M = 200$ antennas and half-wavelength spacing. We use the propagation parameters used in [10, Sec. III.D], which are inspired by the 3rd generation partnership project (3GPP) nonline-of-sight propagation model for 2 GHz carrier frequency [1, A.2.1.1.2-3].

To guarantee 128-bit security, we fix the size of the shares to 128 bits. Furthermore, we consider elliptic curve cryptography (ECC) [7] as encryption algorithm, with the size of the underlying field being roughly twice the security parameter. Thus, we consider key size 256 bits [2], *i.e.*, $b = 256$. For the transmission over the wireless medium, we consider two different coding rates $R = b/n_\mathrm{d} = \{1.28, 0.64\}$, which yield $n_\mathrm{d} = \{200, 400\}$, respectively. We set the length of the pilot sequences n_p to be equal to the number of sensors K. Hence, the coded packets have length $n \in \{250, 450\}$.

The number of available servers in the network is $S = 8$. We consider two possible values of the threshold VAHSS corresponding to t=$\{7, 5\}$. Note that t=7 corresponds to the case where all servers are required to participate in the cloud computation. We shall use as a performance metric the *network availability*, which we define as the fraction of random sensor placements for which the system error probability P_e, averaged over the small-scale fading and the additive noise, is below a given target denoted by $P_{\mathrm{e,target}}$. Mathematically, the network availability is defined as $\eta = \Pr\left[P_\mathrm{e} \le P_{\mathrm{e,target}}\right]$. In the simulation, we set $P_{\mathrm{e,target}} = 10^{-2}$. In Fig. 3, we show the network

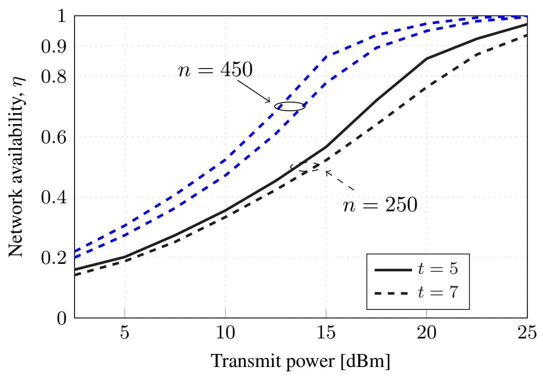

Figure 3: Network availability η for $P_{\mathrm{e,target}}$=10^{-2} as a function of the transmit power ρ [dBm]. The BS, equipped with M=200 antennas, serves a 500 m $\times$ 500 m area where there are K=50 randomly placed sensors. The sensors use coded packets of length n=$\{250, 450\}$ to send their b=256 encrypted bits. Each coded packet contains a preamble of n_{p}=50 pilots symbols, used by the BS to estimate the channel.

availability achievable for the aforementioned system parameters. To interpret the figure, suppose that we desire that our system operates at a network availability equal to 90%. It follows from Fig. 3, that, for n=250 and t=7, the sensors need a transmit power of approximately 24 dBm. If we reduce the threshold parameter to t=5, the required transmitted power can be lowered to 22 dBm. This is expected. Indeed, for a given target error probability $P_{\mathrm{e,target}}$, by reducing the number of t+1 servers needed to operate the protocol, we increase the minimum coded-packet error probability ϵ that needs to be supported over each link according to (3). This translates into a reduction of the required minimum transmit power. As shown in Fig. 3, the required transmit power can be also decreased by increasing the value of n for a fixed n_{p}, i.e., by increasing n_{d}. Indeed, by increasing n_{d}, one can protect the information bits using a stronger, lower-rate channel code, and hence, achieve the target packet error probability using less transmit power. Specifically, for n=450 and t=7, the sensors need to use a transmit power of approximately 17.5 dBm. For n_{d}=400 and t=5, the required transmit power is approximately 15.5 dBm. Note, though, that the latency associated with the wireless transmission of the shares is (roughly) proportional to Sn. So, larger values of n result in larger latencies.

6 RELATED WORK

Privacy-preserving aggregation in sensor networks has received significant attention in the literature [6]. However, existing work has focused on centralized aggregation (*i.e.,* a single server is employed for the aggregation process), while the verifiability property (checking the correctness of the computed aggregation) has been largely ignored. The key novelties of our proposed approach in relation to the state-of-the art are: *(i)* decentralize the aggregation learning, by employing multiple servers; *(ii)* integrate *verifiability* (*i.e.,* proof of correctness of the aggregation process); *(iii)* provide transparency by allowing everyone to check the correctness of the aggregation; *(iv)* achieve high privacy guarantees by avoiding any leakage of information from the sensors' data; and *(v)* provide an analysis of the energy efficiency of the proposed threshold protocol in a 5G wireless sensor network. Previous approaches [9, 12] have studied some of these problems in isolation, without examining the extent of achieving all of them with optimal privacy, accuracy of the aggregation process, and reduced communication cost.

7 CONCLUSIONS

We presented a threshold VAHSS construction that is fault tolerant and provides strong resilience guarantees by allowing a subset of servers not to participate in the cloud-assisted computation of the protocol, whenever decoding errors occur in the wireless communication phase. Our protocol still results in the sum value y and in its corresponding proof of correctness. We proposed an implementation of the protocol within the context of a 5G wireless sensor network, and discussed its performance as a function of the power budget of each sensor. More precisely, for a given power budget and a given latency constraint, we analyzed the so-called network availability, defined as the fraction of sensor placements within a given coverage area, for which the average probability that our threshold VAHSS protocol fails is below a given target.

Acknowledgement. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Alejandro Lancho has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 101024432.

REFERENCES

[1] 3GPP. 2010. *Further advancements for E-UTRA physical layer aspects (Release 9).* Technical Specification (TS). 3rd Generation Partnership Project (3GPP).

[2] Elaine B. Barker, William C. Barker, William E. Burr, W. Timothy Polk, and Miles E. Smid. 2007. *SP 800-57. Recommendation for Key Management, Part 1: General (Revised).* Technical Report. Gaithersburg, MD, USA.

[3] Mihir Bellare and Phillip Rogaway. 1995. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. ACM Press, 62–73.

[4] Emil Björnson, Jakob Hoydis, and Luca Sanguinetti. 2017. Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency. *Foundations and Trends® in Signal Processing* 11, 3-4 (Nov. 2017), 154–655. https://doi.org/10.1561/2000000093

[5] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. 2018. Bulletproofs: Short Proofs for Confidential Transactions and More. In *2018 IEEE Symposium on Security and Privacy (SP).* 315–334. https://doi.org/10.1109/SP.2018.00020

[6] Soroush Abbasian Dehkordi, Kamran Farajzadeh, Javad Rezazadeh, Reza Farahbakhsh, Kumbesan Sandrasegaran, and Masih Abbasian Dehkordi. 2020. A survey on data aggregation techniques in IoT sensor networks. *Wirel. Networks* 26, 2 (2020), 1243–1263. https://doi.org/10.1007/s11276-019-02142-z

[7] Neal Koblitz. 1987. Elliptic curve cryptosystems. *Mathematics of computation* 48, 177 (1987), 203–209.

[8] M.N. Krohn, M.J. Freedman, and D. Mazieres. 2004. On-the-fly verification of rateless erasure codes for efficient content distribution. In *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004.* Berkeley, CA, USA, 226–240.

[9] Tong Li, Chongzhi Gao, Liaoliang Jiang, Witold Pedrycz, and Jian Shen. 2019. Publicly verifiable privacy-preserving aggregation and its application in IoT. *J. Netw. Comput. Appl.* 126 (2019), 39–44.

[10] J. Östman, A. Lancho, G. Durisi, and L. Sanguinetti. 2020. URLLC with Massive MIMO: Analysis and Design at Finite Blocklength. https://arxiv.org/abs/2009.10550

[11] Georgia Tsaloli, Gustavo Banegas, and Aikaterini Mitrokotsa. 2020. Practical and Provably Secure Distributed Aggregation: Verifiable Additive Homomorphic Secret Sharing. *Cryptography* 4, 3 (2020).

[12] Georgia Tsaloli and Aikaterini Mitrokotsa. 2020. Sum It Up: Verifiable Additive Homomorphic Secret Sharing. In *Information Security and Cryptology – ICISC 2019*, Jae Hong Seo (Ed.). Springer International Publishing, Cham, 115–132.

[13] Hailong Yao, Caifen Wang, Bo Hai, and Shiqiang Zhu. 2018. Homomorphic Hash and Blockchain Based Authentication Key Exchange Protocol for Strangers. In *International Conference on Advanced Cloud and Big Data (CBD).* Lanzhou, 243–248.