PrOLoc: Resilient Localization with Private Observers Using Partial Homomorphic Encryption

Amr Alanwar
University of California, Los Angeles
alanwar@ucla.edu

Yasser Shoukry
University of California, Berkeley
University of California, Los Angeles
yshoukry@eecs.berkeley.edu

Supriyo Chakraborty
IBM T. J. Watson Research Lab
supriyo@us.ibm.com

Paul Martin
University of California, Los Angeles
pdmartin@ucla.edu

Paulo Tabuada
University of California, Los Angeles
tabuada@ucla.edu

Mani Srivastava
University of California, Los Angeles
mbs@ucla.edu

ABSTRACT

Aided by advances in sensors and algorithms, systems for localizing and tracking target objects or events have become ubiquitous in recent years. Most of these systems operate on the principle of fusing measurements of distance and/or direction to the target made by a set of spatially distributed observers using sensors that measure signals such as RF, acoustic, or optical. The computation of the target's location is done using multilateration and multiangulation algorithms, typically running at an aggregation node that, in addition to the distance/direction measurements, also needs to know the observers' locations. This presents a privacy risk for an observer that does not trust the aggregation node or other observers and could in turn lead to lack of participation. For example, consider a crowd-sourced sensing system where citizens are required to report security threats, or a smart car, stranded with a malfunctioning GPS, sending out localization requests to neighboring cars – in both cases, observer (i.e., citizens and cars respectively) participation can be increased by keeping their location private. This paper presents PrOLoc, a localization system that combines partially homomorphic encryption with a new way of structuring the localization problem to enable efficient and accurate computation of a target's location without requiring observers to make public their locations or measurements. Moreover, and unlike previously proposed perturbation based techniques, PrOLoc is also resilient to malicious active false data injection attacks. We present two realizations of our approach, provide rigorous theoretical guarantees, and also compare the performance of each against traditional methods. Our experiments on real hardware demonstrate that PrOLoc yields location estimates that are accurate while being at least 500× faster than state-of-art secure function evaluation techniques.

CCS CONCEPTS

•**Computer systems organization** →**Embedded systems;** *Redundancy;* Robotics; •**Networks** →Network reliability;

IPSN 2017, Pittsburgh, PA USA

DOI: http://dx.doi.org/10.1145/3055031.3055080

KEYWORDS

Secure localization, privacy, homomorphic encryption, Paillier cryptosystem

ACM Reference format:
Amr Alanwar, Yasser Shoukry, Supriyo Chakraborty, Paul Martin, Paulo Tabuada, and Mani Srivastava. 2017. PrOLoc: Resilient Localization with Private Observers Using Partial Homomorphic Encryption. In *Proceedings of The 16th ACM/IEEE International Conference on Information Processing in Sensor Networks, Pittsburgh, PA USA, April 2017 (IPSN 2017),* 12 pages.
DOI: http://dx.doi.org/10.1145/3055031.3055080

1 INTRODUCTION

Advances in localization techniques have enabled a multitude of services including navigation, targeted advertisements, and location-aware applications. With the growing prevalence of powerful mobile computing devices, techniques for localization both in outdoor and indoor environments are becoming both more wide-spread and more varied. Many of these localization techniques depend on analyzing various *observations* of a distinct phenomenon or event—an acoustic chirp, an optical signature, an RF signal, etc—in order to infer the location of some object or event whose location is unknown and possibly changing over time. In such scenarios, measured signals are often captured by a number of sensors or *observers* and then typically communicated to a centralized computer or an *aggregator* for aggregate post-processing and eventually location estimation. For example, cellular signals from mobile devices can be measured by base stations to infer the location of the device based on geometric multilateration, and wireless sensor nodes can use microphones along with beamforming algorithms to localize the source of specific acoustic signatures. The final location estimate is then sent to the interested party who initiated the *inquiry*.

Information Leakages During Localization: Traditionally, the infrastructure required to measure and observe the various signals required for localization—microphones, radios, light sensors, etc.—are treated as non-private entities whose locations and measurements are either known or easily inferred. Knowing both the locations and the measurements of these devices is required for localization, and thus privatizing this information on a centralized computer has significant consequences on the localization process itself. While in many cases the privacy of the observers' locations and measurements is of little concern—for example WiFi routers

Amr Alanwar and Yasser Shoukry are equally contributing authors.

in a public shopping mall—in other cases publicizing this data can have serious security implications.

Information leakage typically occurs either when (1) the observers do not belong to the same *trust* zone. For example, consider a smart car, that has lost its GPS signal and is sending out requests for its localization [23]. The neighboring cars (and their owners) receiving the request may not belong to the same trust zone (e.g., defined by members of the same organization) as the smart car, and thus may not be willing to share their location information without privacy guarantees. Similarly, in the context of smart cities, citizens are often required to report safety and maintenance issues to government authorities. In such settings, it is not uncommon for citizens to not completely trust each other or the government. (2) The observers all belong to the same trust zone, but not the cloud-based server that is used to perform the localization. For example, in a battlefield, soldiers often monitor and report the location of nearby events. While all the soldiers, of a unit, belong to the same trust zone, the cloud/aggregator service could be outside this trust zone and has a risk of being compromised. Therefore, performing localization while preserving the privacy of the observers is highly desirable. In fact, from the above examples, we find that observer data leakage can occur through one of the following ways:

Malicious Aggregators: To perform a joint position estimation, given the information measured by observers, the observer locations as well as the observations themselves are typically stored at a remote server. In multilateration and multiangulation systems, for example, infrastructure positions, ranges, and angles are often communicated to a single device which then performs an optimization routine in order to arrive at an accurate position estimate of some target. Upon receiving the sensitive observers' data, a malicious aggregator could relay this information to interested third parties with more nefarious intents. In many scenarios, this information can be incredibly sensitive—for example, military signal towers used to locate soldiers in the field, distributed and crowdsourced detection of gunshots or other illicit activities for which the observers should remain anonymous in order to be protected, etc.

Aggregator Information Leaks: The aggregation and centralization of the observers' information creates a point of weakness in traditional systems, where even a *trusted* aggregator may inadvertently leak sensitive data—e.g. the infamous examples of hackers gaining root access to the WordPress server [4] and the data breach of customers' information on JCPenny servers [2]. Distributing the estimation of the unknown target location across multiple servers or across the observers themselves can alleviate this security risk to some degree, but where distributed processing is infeasible and in order to eliminate the risk rather than merely reducing it, observer locations and measurements should never be disclosed to a centralized server.

Challenges in Performing Private Localization: To address the privacy requirements and prevent leakage of sensitive observer information, recently researchers have begun to re-imagine localization techniques in a more privacy-aware and privacy-preserving manner. We identify below several challenges that need to be addressed for any realizable solution:

- **Accuracy of localization:** The accuracy of the localization process is key for its use in various applications (e.g., navigation, tracking gunshots, monitoring objects and so on).
- **Energy-constrained observers:** The observers providing the measurements are often energy-constrained miniature devices. Thus, any solution providing strong privacy guarantees should also be sufficiently efficient in terms of both computation and communication costs.
- **Privacy preserving:** while location information is, in general sensitive, there should be protection against passive adversaries eavesdropping on the protocol messages and trying to infer sensitive observer information.
- **Resilience under active attacks:** The privacy guarantees provided by any solution should be resilient to *collusion attacks* and *false data injection attacks* from active adversarial observers. As part of a collusion attack, a group of observers can collude between themselves and/or with the aggregator node to reveal the sensitive information of other participating observers. Similarly, malicious observers can attempt to degrade the performance of the localization process itself by reporting false noisy measurements to the aggregator node.

This paper presents *PrOLoc*—a set of novel protocols and algorithms for estimating the location of a given target based on measurements from observers. More specifically, algorithms in PrOLoc are designed to leverage partially homomorphic encryption techniques in order to calculate a location estimate based on encrypted range estimates from observers with encrypted locations. The resulting (encrypted) estimate is then communicated to the inquiring party such that only it can decrypt it and, furthermore, in such a way that even *it* cannot infer the private locations or ranges of any given observer – providing strong privacy and security guarantees.

Our technique by design produces negligible degradation in localization accuracy. An immediate consequence of this fact is that we can further augment our technique with residue checking based schemes to incorporate resilience against active attacks. We implemented it on a real-time testbed consisting of four observers to demonstrate its feasibility on energy-constrained devices. In summary, the contributions of this paper are as follows:

- We present *PrOLoc*—a method for performing localization of a mobile target based on encrypted measurements from private observers. PrOLoc leverages the Paillier additive homomorphic cryptosystem to efficiently estimate target positions in real time without revealing the locations or measurements of any given observer.
- We analyze the performance of two localization algorithms in terms of computational efficiency and privacy preservation: (i) polyhedra-based localization: a partially homomorphic implementation of least squares, range-based localization and (ii) polyhedra-based localization using alternating projection: a partially homomorphic implementation of an alternating projection algorithm.
- We evaluate the results of the above secured localization algorithms using an end-to-end system of real range measurement hardware in a laboratory-scale room, analyzing both the accuracy and the efficiency of each algorithm for a scaled representation of a typical deployment.

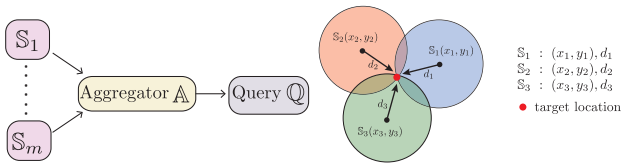


Figure 1: (left) Communication flow for Aggregator-based based localization. (right) Trilateration using ranges from three observers ($\mathbb{S}_i$)

- We provide strong theoretical guarantees regarding observer privacy and resilience for each of the demonstrated algorithms.

The rest of this paper is organized as following. Section 2 formulates the definition of the privacy and the resilience. A review on related work in literature is shown in Section 3. In Section 4, we provide the mathematical preliminaries and basic cryptographic blocks. The main contributions of this paper are Sections 5 and 6 where we discuss novel localization algorithms. The resilient privacy-aware localization is introduced in 7. The proposed algorithms are evaluated using a real-time testbed in Section 8. Finally, we concludes the paper and show the limitations in Section 9.

2 PROBLEM SETUP

In what follows, we describe the different entities involved in the localization process as well as the notions of privacy that our proposed algorithms are interested in achieving.

2.1 Entities

The localization process involves the following entities:

- **Observers (or Sensors) $\mathbb{S}_i$:** Entities that make some measurement of the target that is then used for localization. In this paper, these measurements are range estimates, as shown in Fig. 1 (right). We assume that we have m observers (sometimes referred to as Sensors or anchors). Each observer $\mathbb{S}_i$ (with i being the index of the observer, i.e. $i \in \{1, \ldots, m\}$) possesses three sensitive data $\mathbb{S}_i = (x_i, y_i, d_i)$, where $(x_i, y_i) \in \mathbb{R}^2$ denotes the x-y position of the ith observer in 2-dimensional space[1] and $d_i \in \mathbb{R}$ denotes the measured distance between the observer and a target $\mathbb{T}$ (see below). The objective of the proposed protocols in this paper is to ensure the privacy of all observer locations (x_i, y_i) as well as the distance d_i.
- **Malicious Observers (or Sensors) $\mathbb{S}_i$:** Adversarial observers who launch false data injection attacks by reporting false observer locations (x_i, y_i) and/or false distance d_i.
- **Target $\mathbb{T}$:** A passive entity whose location $z_{\mathbb{T}} = (x_{\mathbb{T}}, y_{\mathbb{T}})$ needs to be computed using the sensitive information possessed by the observers.
- **Aggregator $\mathbb{A}$:** An *untrusted party* which has a known public key $\text{pk}_{\mathbb{A}}$ and a private key $\text{sk}_{\mathbb{A}}$. Upon receiving all encrypted information sent by observers, it applies a localization algorithm in order to calculate the (encrypted) target location.
- **Query Node $\mathbb{Q}$:** An *untrusted party* that has a known public key $\text{pk}_{\mathbb{Q}}$ and a hidden private key $\text{sk}_{\mathbb{Q}}$. The query node is the only node that is entitled to know the target location. This query node can be physically the same as the target, or it can be another entity entirely (other than the aggregator, in order to preserve privacy).

[1]For the sake of simplicity we focus on localization in 2-dimensional space, although the described algorithms can be naturally extended to localization of objects in 3-dimensional space.

2.2 Attacker Model

We focus on both active (false data injection) attacks as well as passive (privacy-leaking) attacks. That is, we assume that observers truthfully report their measured distances and that the aggregator is following the localization algorithms correctly with the exception that malicious observers may adversarially send false data about their locations and/or distance to the target.

2.3 Privacy Definitions

We define our privacy notions with the intuition that each entity involved in the localization algorithm should learn nothing about the sensitive information possessed by the other entities even if some of them collude together. However, we note that there exists some **immutable privacy leak** whenever the query node is involved. This can be defined as follows:

Immutable privacy leak: After each run of any localization protocol, the query node learns that all observers lie inside a circle whose center is the target location $z_{\mathbb{T}} = (x_{\mathbb{T}}, y_{\mathbb{T}})$ and whose radius is the maximum sensing range (or RF communication range) of the observers. With this definition of *immutable privacy leaks*, we can define our privacy goals as follows:

- **Observer Obliviousness:** By the end of multiple runs of protocol execution, if all but one observers collude—by exchanging their private values—the coalition should learn nothing about the remaining observer.
- **Aggregator Obliviousness:** By the end of multiple runs of protocol execution, if the aggregator $\mathbb{A}$ colludes with all observers but one, the coalition learns nothing about the remaining observer.
- **Non-colluding Aggregator Obliviousness:** A protocol satisfies the non-colluding aggregator obliviousness property if after multiple runs of the protocol the aggregator $\mathbb{A}$ learns nothing about the observers' sensitive information.
- **Query Node Obliviousness:** By the end of multiple runs of protocol execution, if the query node $\mathbb{Q}$ colludes with all observers but one, the coalition learns nothing about the remaining observer other than what is implied by the *immutable privacy leak*.

The traditional privacy notion of a "zero knowledge protocol" [18] considers a protocol secure if the adversarial agents do not leak any information other than what is implied by the final output (the computed target location obtained by the Query node in our problem). It is crucial to note that the previous privacy definitions ask for more stringent requirements in the sense that they require that even the final output should not leak any information about the sensitive information.

Note that in all the previous privacy notions the aggregator $\mathbb{A}$ and query node $\mathbb{Q}$ are not allowed to combine their respective data in any malicious manner to infer more about the observers. This constraint is implicitly assumed in many privacy-preserving techniques like differential privacy in which a malicious aggregator which does not corrupt the data correctly lead to hindering the privacy of the system. In practice, such constraint is implemented by natural division of labor. Examples of which appears in multi-institution financial data analysis where an organization aggregates sensitive financial data gathered from multiple competing financial intuitions and trusted not to collude with any of them [5]. Because of this natural division of labor and this very typical publish-subscribe model, we can safely assume for the sake of this paper that the two parties—querier and aggregator—operate under different authorities and are disallowed to collude

2.4 Utility and Performance Measures

The accuracy of localization is the key for its use in mission-critical systems. In addition, the observer nodes are often energy-constrained sensors. To accommodate the above practical constraints while still maintaining the desired privacy and resilience level we define the following utility metrics:

- **Localization Error:** is defined as the distance between the actual target location (ground truth) and the calculated (or estimated) target location.
- **Execution Time:** is defined as the time needed by the aggregator to fuse the measurements and produce the final estimate of the target.
- **Observer Communication Cost:** is defined as the total number of bits sent and received by all observers.

2.5 Resilience Definition

Under a strict minority of malicious active adversaries, we define *resilience* as negligible increase in the overall localization error compared to when the no active adversaries are present. The increased communication cost and execution time, incurred to achieve protocol resilience should also be negligible.

We now provide a brief overview of prior work and comment on the suitability of existing techniques to our problem setting.

3 COMPARISON TO PRIOR WORK

A closely related problem to localization with private observations is that of secure data aggregation either in a distributed setting or in the presence of an untrusted aggregator. We classify the various techniques that have been proposed for the above problem into two broad categories and discuss why they are not suitable for our problem setting.

3.1 Differential Privacy Techniques

These techniques rely on the addition of structured noise to the data before sharing with the aggregator such that the aggregate data preserves the privacy of each individual datum. Differential privacy [12, 13] provides rigorous privacy guarantee by requiring the output of the aggregator to not change significantly even if an observer has opted out of the data collection. More recently, variant schemes such as local differential privacy [36] and Geo-Indistinguishability [6] have been designed to ensure differential privacy for location data. However, to achieve these guarantees in a distributed setting without a trusted aggregator, each observer needs to add noise to make their own data differentially private. This results in an aggregated noise that far exceeds the required amount to ensure differential privacy for the aggregated result, severely degrading the accuracy of the localization result and making it unsuitable for use in critical systems.

To overcome the addition of excessive noise, a combination of homomorphic encryption strategies with distributed noise generation have been proposed. In [32], each observer generated his share of the aggregate noise required for differential privacy [14] and sent encrypted, obfuscated data to the aggregator. Similarly in [32], homomorphic encryption is used to secure shared data and the encryption key is generated using the current iteration number and the originally established key. Differential privacy of the aggregated data is established using noise drawn from a symmetric geometric distribution. The above techniques are specifically designed for simple aggregation (mean computation) using collected data whereas in our case the localization process requires more sophisticated operations making these noise generation strategies inapplicable.

3.2 Secure Multi-Party Computation

A multi-party computation protocol is secure if no information about the private data held by the parties can be inferred from the messages exchanged during the execution of the protocol. The only information that can be inferred about the private data is that which could be inferred from the output of the overall function alone [27, 38]. This makes secure multiparty computation (SMC) an excellent candidate for secure aggregation without requiring an aggregator. **A comparative study on using SMC for secure aggregation can be found in [19] where again the aggregation of the data is limited to performing summation operations, which, as noted before, is insufficient for localization purposes**. Furthermore, Yao's work on garbled circuits [38], has resulted in the construction of general-purpose SMC circuits [22], that can compute any arbitrary polynomial function over encrypted data. However, these general-purpose black box techniques, are not only complicated to implement but also very slow – requiring multiple rounds of message exchanges – and significantly increasing the communication cost of the observer. We made these observations not only from our own experimentation with these techniques and similar conclusions have also been noted in [31].

3.3 Fully Homomorphic Encryption (FHE)

FHE [16] has recently been used for countering server-side information leakages. Over the past few years, significant work (in the form of an FHE library [17]) has gone into making it practical, and it has been used for computationally expensive tasks over genome data [25] and recently for classification over encrypted data [7]. Also, prior work achieves less than a second per simple operation [11]. However, the resulting scheme is still impractical for real-time localization services owing to high latencies [37] as reported in our experiments in Section 8.

3.4 Privacy Through Selective Obfuscation

One common technique to preserve privacy is through obfuscation via addition of noise to specific dimensions of sensor signal such that, when aggregated, the noise cancels out (i.e. sums to zero) in the dimension of interest. Specifically in the context of localization, such intermediate data obfuscation has been formulated in [28, 33], where each node generates random noise which is then added to its private input. Unfortunately, this technique assumes a non-adversarial setup where the observer nodes are always fully compliant and follow the designed localization protocol. However, in case of adversarial nodes that do not add the right amount of noise, and/or colluding nodes that reveal their noise values, the localization accuracy and the privacy guarantees can be significantly degraded, violating our model requirements (as defined in Section 2.2).

3.5 Secure localization

Secure localization in case of non-coordinated attacks, where compromised nodes independently modify the measured distance to prevent accurate localization of the target, was presented in [15]. Similarly, SeRLoc [26] demonstrated defense mechanisms against wormhole [21] and Sybil attacks [10]. Secure RSS-based localization was introduced in [9] and a scheme for secure Time-of-Arrival based localization was presented in [34]. However, the above techniques did not in particular consider the privacy of the observers participating in the localization process.

4 PRELIMINARIES

4.1 Mathematical Notation

We denote by $\mathbb{R}, \mathbb{Z}$ the set of real numbers and integers, respectively. For a vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, we denote by $x(i)$ the ith element of the vector. Similarly, for a matrix $A \in \mathbb{R}^{n\times n}$ we denote by $A(i) \in \mathbb{R}^{1\times n}$ the ith row of the matrix A. We also denote by A^T and A^{-1} the transpose and the inverse of the matrix $A \in \mathbb{R}^{n\times n}$. Finally, we denote by $\|x\|_2$ the 2-norm of the vector $x \in \mathbb{R}^n$ which is equal to $\|x\|_2 = \sqrt{\sum_{i=1}^{n} x(i)^2}$.

4.2 Paillier homomorphic cryptosystem

Our protocols make heavy use of a particular homomorphic cryptosystem named Paillier additive homomorphic cryptosystem [30]. This is a probabilistic public key cryptography scheme that allows two important operations, namely (i) addition of two encrypted values and (ii) multiplication of an encrypted value by a plaintext value. That is, if we denote by $[\![a]\!]_{\text{pk}}$ the encryption of a using the public key pk then the Paillier cryptosystem supports two operations $\oplus$ (and accordingly $\ominus$) and $\otimes$ such that:

$$\textsc{Decrypt}_{\text{sk}}([\![a]\!]_{\text{pk}} \oplus [\![b]\!]_{\text{pk}}) = \textsc{Decrypt}_{\text{sk}}([\![a+b]\!]_{\text{pk}}) = a + b$$
$$\textsc{Decrypt}_{\text{sk}}(a \otimes [\![b]\!]_{\text{pk}}) = \textsc{Decrypt}_{\text{sk}}([\![a\times b]\!]_{\text{pk}}) = a \times b$$

where sk is the private key associated with the public key pk. Note that these two properties can be generalized to perform multiplication between a plaintext matrix and a vector of encrypted variables. That is given a vector $x \in \mathbb{Z}^n$ and a matrix $M \in \mathbb{Z}^{n\times n}$, we can compute $M \otimes [\![x]\!]_{\text{pk}}$ as:

$$M \otimes [\![x]\!]_{\text{pk}} = \begin{bmatrix} M(1,1) \otimes [\![x(1)]\!]_{\text{pk}} \oplus \ldots \oplus M(1,n) \otimes [\![x(n)]\!]_{\text{pk}} \\ \vdots \\ M(n,1) \otimes [\![x(1)]\!]_{\text{pk}} \oplus \ldots \oplus M(n,n) \otimes [\![x(n)]\!]_{\text{pk}} \end{bmatrix}$$

where, with some abuse of notation, we used $\otimes$ in $M \otimes [\![x]\!]_{\text{pk}}$ to denote the multiplication of a plaintext matrix M with an encrypted vector $[\![x]\!]_{\text{pk}}$. The security guarantees of the Paillier cryptosystem rely on a standard cryptographic assumption named Decisional Composite Residuosity (DCR) [30].

4.3 Secure Comparison Protocol

The second basic block of our protocols is comparison over encrypted data. In this protocol, if party A has a set of encrypted data $[\![a_1]\!]_{\text{pk}_B}, \ldots, [\![a_m]\!]_{\text{pk}_B}$ encrypted using the Paillier cryptosystem using B's public key, then by the end of this protocol, entity B knows the index of the maximum (or minimum) value, i.e. B learns $\arg\max\{a_1, \ldots, a_m\}$, and nothing else whenever both A and B are honest-but-curious. Different instantiations of this protocol are proposed in literature and are either based on data obfuscation [7] or on Garbled Circuits [20]. We call this *secure comparison* protocol $\textsc{SecCompare}([\![a_1]\!]_{\text{pk}_B}, \ldots, [\![a_m]\!]_{\text{pk}_B})$.

4.4 Traditional Least Squares Localization

The objective of any localization algorithm is to calculate a target's position, denoted by $(x_{\mathbb{T}}, y_{\mathbb{T}})$, using all information provided by the sensors $\mathbb{S}$. One common method of predicting the target position is by using range estimates between sensors and target to constrain the target's position on a 2D plane. Towards this end, we consider the geometric multilateration method which calculates the target location based on the geometry imposed by sensor-target range estimates which can be casted as the solution of the following least squares optimization problem:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \arg \min_{\hat{z}_{\mathbb{T}}=(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}})\in\mathbb{R}^2} \|A_{\mathbb{S}}\hat{z}_{\mathbb{T}} - b_{\mathbb{S}}\|_2^2 \quad (1)$$

$$A_{\mathbb{S}} = \begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) \\ \vdots & \vdots \\ 2(x_m - x_1) & 2(y_m - y_1) \end{bmatrix},$$

$$b_{\mathbb{S}} = \begin{bmatrix} x_2^2 + y_2^2 - d_2^2 - (x_1^2 + y_1^2 - d_1^2) \\ \vdots \\ x_m^2 + y_m^2 - d_m^2 - (x_1^2 + y_1^2 - d_1^2) \end{bmatrix}.$$

Note that we use the subscript $\mathbb{S}$ in $A_{\mathbb{S}}, b_{\mathbb{S}}$ to emphasize the fact that both the matrix $A_{\mathbb{S}}$ as well the vector $b_{\mathbb{S}}$ depend on the (potentially sensitive) sensors' information $\mathbb{S}$. This will play a vital role in designing our algorithms as we will show in the next Section. The minimum of this optimization function (and hence the target location) can be obtained as:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = (A_{\mathbb{S}}^T A_{\mathbb{S}})^{-1} A_{\mathbb{S}}^T b_{\mathbb{S}} \quad (2)$$

Calculating the target location using (2) requires both multiplication and addition of sensitive information. While this cannot be implemented directly using only additive homomorphic encryption like Paillier algorithm, it can be implemented using other Secure

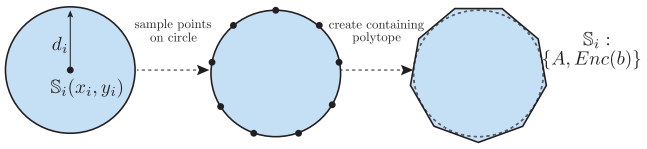


Figure 2: Discretizing the circle as a polyhedron

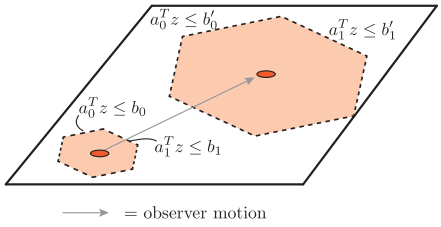


Figure 3: Polyhedron discretization of range circle under translation and scaling (varying b_i and fixed a_i).

Function Evaluation (SFE) techniques like Fully Homomorphic Encryption (FHE) [16], Garbled Circuits (GC) [22], or hybrid SFE (partial homomorphic encryption + Garbled circuits) [20]. As shown by our experiments (Section 8) and other researchers' work, FHE is still far from being useful, and GC and hybrid SFE requires large number of communication rounds between observers and aggregator which violate our utility measure (observer communication cost). Rethinking the optimization problem (1) from a privacy point of view, we introduce in the next two sections new formulations of the localization optimization problem that can be implemented using only partial homomorphic encryption (e.g. Pailliaer cryptosystem) and do need a large number of communication rounds.

5 THE POLYHEDRA-BASED LOCALIZATION ALGORITHM

In this section, we introduce our first contribution of this paper: a re-formalization of the localization algorithm which requires only *additive* homomorphic encryption rather than fully homomorphic encryption or garbled circuits. Additionally, this re-formalization preserves the strong privacy guarantees. We call this localization algorithm *polyhedra based localization*, and we describe it in detail in this section. We focus in this section on the case when no false data injection attack is present and concentrate only on the problem estimating the location of the target node in a privacy-preserving manner. In Section 7, we show how to augment the proposed privacy preserving protocols to achieve resilience against false data injection attacks.

5.1 Polyhedra Based Localization

The intuition behind the *polyhedra based localization* algorithm is as follows. Traditional geometry-based localization methods like the least squares approach discussed above treat range measurements as an estimate that the target $\mathbb{T}$ lies on some circle whose radius is equal to the estimated distance and whose center is the coordinate of the sensor. In essence, communicating this constraint requires communicating both radius (d_i) and center ($x_{\mathbb{S}_i}, y_{\mathbb{S}_i}$), exposing sensitive information about the sensors themselves. Rather than parameterizing the sensing range as a circle, we seek another parameterization that can still represent the estimated range without sacrificing sensor privacy. Towards this end, we propose using polyhedra to represent the sensed range. As shown in Figure 2, we can obtain such polyhedra by randomly sampling the circumference of the circle representing the measured range. Recall that a polyhedron is the intersection of f halfspaces and can be represented using their unit normal vectors and offsets as:

$$\mathcal{P} = \{z = (x, y) \,|Az \leq b\} \quad \text{where} \quad A = \begin{bmatrix} a_1^T \\ \vdots \\ a_f^T \end{bmatrix}, b = \begin{bmatrix} b_1 \\ \vdots \\ b_f \end{bmatrix}$$

Since any polyhedron is parameterized by the matrix A and the vector b, we use the following shorthand notation: $\mathcal{P}(A, b)$. An intrinsic characteristic of polyhedra is the following: **For a polyhedron $\mathcal{P}(A, b)$, the matrix A (which is the collection of the normal vectors of the facets) represents only the shape of the polyhedron, specifying neither the location nor the scale of the polyhedron.** Figure 3 shows an example of this geometric fact. In Figure 3 we show two different polyhedra that have exactly the same matrix A (same normal vectors to facets) but different b vectors. These two polyhedra have different locations and sizes as well. Hence, we use the notation A_i and $b_{\mathbb{S}_i}$ to denote the polyhedron constructed by the ith observer where the subscript $\mathbb{S}_i$ in $b_{\mathbb{S}_i}$ is used to stress the fact that the sensitive information of the observer is encoded inside the vector $b_{\mathbb{S}_i}$ and not in the matrix A_i.

Given a set of polyhedra from different observers, our objective is to fuse them in order to calculate the target location. We note that the target lies in the intersection of all of the m polyhedra constructed by observers. This intersection creates another polyhedron (denoted by $\mathcal{P}_{\mathbb{T}}$) that can be represented as the intersection of all facets of all the m polyhedra. That is:

$$\mathcal{P}_{\mathbb{T}} = \{z = (x, y) \in \mathbb{R}^2 \mid \overline{A}z \leq \overline{b}_{\mathbb{S}}\} \text{ where } \overline{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix}, \overline{b}_{\mathbb{S}} = \begin{bmatrix} b_{\mathbb{S}_1} \\ \vdots \\ b_{\mathbb{S}_m} \end{bmatrix}$$

Since the target location lies inside the polyhedron $\mathcal{P}_{\mathbb{T}}$, we can calculate the target location by solving the following optimization problem:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \arg \min_{(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) \in \mathbb{R}^2} \left\| \overline{A}\hat{z}_{\mathbb{T}} - \overline{b}_{\mathbb{S}} \right\|_2^2 \tag{3}$$

Such a solution can be calculated as:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \left(\overline{A}^T \overline{A}\right)^{-1} \overline{A}^T \overline{b}_{\mathbb{S}} \tag{4}$$

The major difference between the solution (4) in the polyhedra-based algorithm and the corresponding one in the standard least squares one (2) is the fact that the matrix $\overline{A}$ does not hold any sensitive information and hence can be communicated as plaintext compared to the necessity of encrypting the matrix $A_{\mathbb{S}}$ in the standard least squares algorithm. Hence, by reducing the encryption

requirement to only encrypting the vector $\overline{b}_{\mathbb{S}}$, our polyhedra-based algorithm is able to use additive homomorphism rather than full homomorphism. This in turn results in a much more efficient algorithm in terms of execution time as reflected by the experimental results shown in Section 8. The details of the protocol are given in the next subsection.

5.2 Poly-LSQ: Polyhedra-based Least Squares Localization

5.2.1 `Poly-LSQ.Encrypt`. Since we assume that the shape of each polyhedron is fixed, we can assume without loss of generality that the matrix A is known to all parties and is not required to be communicated. Hence, we focus on encrypting the vectors $\overline{b}_{\mathbb{S}_i}$ for all sensors $i \in \{1, \ldots, m\}$. For each facet, the ith observer encrypts the corresponding offset vector $\overline{b}_{\mathbb{S}_i}$ using the public key of the query node $\text{pk}_{\mathbb{Q}}$ followed by another encryption using the public key of the aggregator $\text{pk}_{\mathbb{A}}$ resulting in the following message content:

$$msg_i = (\llbracket \llbracket \overline{b}_{\mathbb{S}_i} \rrbracket_{\text{pk}_{\mathbb{Q}}} \rrbracket_{\text{pk}_{\mathbb{A}}}) \quad \forall i \in \{1, \ldots, m\} \tag{5}$$

Note that the purpose of the second encryption is to prevent the query node from accessing $\llbracket \overline{b}_{\mathbb{S}_i} \rrbracket_{\text{pk}_{\mathbb{Q}}}$ which it could then decrypt using its own private key $\text{sk}_{\mathbb{Q}}$. Hence, this second encryption does not need to be carried out using a homomorphic cryptosystem but rather any public key cryptosystem. For sake of simplicity, and since the Paillier cryptosystem is indeed a public key cryptosystem, we use the same notation for this second encryption.

5.2.2 `Poly-LSQ.Aggregate`. Once all observers have sent their messages to the aggregator $\mathbb{A}$, the aggregator constructs the matrix $\tilde{A}$ by stacking all the matrices A together. Similarly, it uses its secret key $\text{sk}_{\mathbb{A}}$ to decrypt $\llbracket \llbracket \overline{b}_{\mathbb{S}_i} \rrbracket_{\text{pk}_{\mathbb{Q}}} \rrbracket_{\text{pk}_{\mathbb{A}}}$ and constructs the vector $\llbracket \overline{b}_{\mathbb{S}} \rrbracket_{\text{pk}_{\mathbb{Q}}}$, i.e.,

$$\overline{A} = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix}, \quad \llbracket \overline{b}_{\mathbb{S}} \rrbracket_{\text{pk}_{\mathbb{Q}}} = \begin{bmatrix} \llbracket \overline{b}_{\mathbb{S}_1} \rrbracket_{\text{pk}_{\mathbb{Q}}}, \\ \vdots \\ \llbracket \overline{b}_{\mathbb{S}_m} \rrbracket_{\text{pk}_{\mathbb{Q}}} \end{bmatrix}$$

In the next step, $\mathbb{A}$ computes the encrypted value of the target estimate $(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}})$ as:

$$(\hat{x}_{\mathbb{T}}, \hat{y}_{\mathbb{T}}) = \left(\overline{A}^T \overline{A}\right)^{-1} \overline{A}^T \otimes \llbracket \overline{b}_{\mathbb{S}} \rrbracket_{\text{pk}_{\mathbb{Q}}}$$

Thanks to the polyhedra representation, the matrix $\overline{A}$ is plaintext and hence the matrix $(\overline{A}^T\overline{A})^{-1}\overline{A}^T$ is computed on plaintext before using Paillier algorithm to multiply it by the encrypted vector $\llbracket \overline{b}_{\mathbb{S}} \rrbracket_{\mathbf{pk}_{\mathbb{Q}}}$.

5.2.3 `Poly-LSQ.Decrypt`. Finally, the aggregator $\mathbb{A}$ sends the computed estimate $\llbracket \hat{z}_{\mathbb{T}} \rrbracket_{\text{pk}_{\mathbb{Q}}}$ to the query node $\mathbb{Q}$. The query node decrypts the message using its private key $\text{sk}_{\mathbb{Q}}$ to retrieve the final estimate $\hat{z}_{\mathbb{T}}$.

5.2.4 Privacy Analysis of the Poly-LSQ Protocol. Finally, we can show that the proposed Poly-LSQ protocol satisfies the strong privacy notions. This is summarized in the following theorem.

> Theorem 5.1. *Assume that all entities are honest-but-curious and under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the Poly-LSQ localization protocol ensures (i) observer obliviousness, (ii) aggregator obliviousness, and (iii) query node obliviousness.*

For space constraints we omit the formal proof of the previous theorem and provide below a brief sketch of the proof.

Proof sketch: In this protocol, the observer and aggregator obliviousness follow from the data being double encrypted by the public keys $\text{pk}_{\mathbb{Q}}$ and $\text{pk}_{\mathbb{A}}$. The query node obliviousness follows from the following facts (1) revealing the $\overline{A}$ matrix does not lead to any privacy leakage as shown before (2) there exist infinitely many values for the observer sensitive information that leads to the same final target estimate. Algebraically, let $H = \left(\overline{A}^T\overline{A}\right)^{-1}\overline{A}^T \in \mathbb{R}^{2\times mf}$ and note that the relation between the released target location $\hat{z}_{\mathbb{T}} \in \mathbb{R}^2$ and the sensitive information $\overline{b}_{\mathbb{S}} \in \mathbb{R}^{mf}$ is captured by the following linear equations $\hat{z}_{\mathbb{T}} = H\overline{b}_{\mathbb{S}}$. Assume without loss of generality that the query node colludes with the observers $\mathbb{S}_1, \ldots, \mathbb{S}_{m-1}$ in order to get the vector $\overline{b}_{\mathbb{S}\setminus\mathbb{S}_m}$. We can then repartition the matrix H into two parts $H' \in \mathbb{R}^{2\times(m-1)f}$ and $H'' \in \mathbb{R}^{2\times f}$ such that:

$$\hat{z}_{\mathbb{T}} = H'\overline{b}_{\mathbb{S}\setminus\mathbb{S}_m} + H''\overline{b}_{\mathbb{S}_m} \Leftrightarrow y = H''\overline{b}_{\mathbb{S}_m}$$

where $y = \hat{z}_{\mathbb{T}} - H'\overline{b}_{\mathbb{S}\setminus\mathbb{S}_m}$ is the vector of all information that the coalition posses. It follows from the dimension of the matrix H'' that the vector y has 2 elements while the vector of the unknowns $\overline{b}_{\mathbb{S}_m}$ has f unknowns. Therefore, for the worst case when f is equal to 3 (the minimal number of facets that can be used to construct a polyhedra), then we have unknown variables more than the number of equations. Therefore there exists infinitely many solutions to the equation $y = H''\overline{b}_{\mathbb{S}_m}$ and we conclude the privacy of the proposed algorithm.

5.3 Geometric Dilution Of Precision

Geometric Dilution Of Precision (GDOP) refers to the degradation of the localization performance (measured by the error between the actual target location and the calculated one) due to geometrical arrangement amongst the range vectors between the receiver and the observers. This phenomenon is widely observed in many localization systems including GPS-based localization depending on the geometric positioning of GPS satellites (which represent the observers in our problem setup) compared to the actual target position [24]. Our experiments shown in Section 8 reveal that the proposed polyhedra-based least squares algorithm is sensitive to geometric positioning of observers. To better illustrate the sensitivity to the observers' geometric configuration, Figures 4(a) and 4(b) show two cases where the observers' locations are fixed while the target location is (i) inside the convex hull of the observers (Figure 4(a)) and (ii) outside the convex hull of the observers (Figure 4(b)). In each case, we plot the entire intersection polyhedron as well as the value of the objective function over the entire space. As shown in Figure 4(a), the center of the intersecting polyhedron $\mathcal{P}_{\mathbb{T}}$ lies exactly at the target location and hence the minimum of the objective function also lies at the target location. On the other

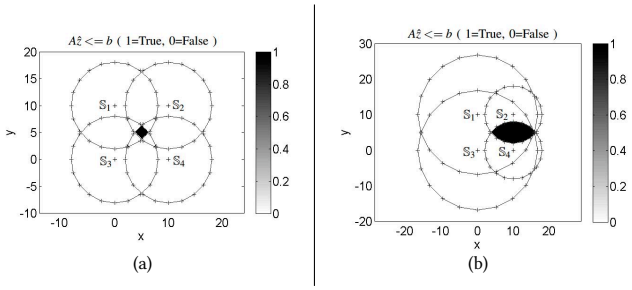


Figure 4: Different geometric configurations showing the size of the intersecting polyhedron and hence the geometric dilution of precision of the Poly-LSQ protocol. We plot two cases (a) a target located at $x = 5, y = 5$ within the observers' convex hull and (b) a target located at $x = 15, y = 5$ outside the observers' convex hull.

hand, when the target location is outside the convex hull of the observers, the center of the intersecting polyhedron $\mathcal{P}_{\mathbb{T}}$ is away from the target location and similarly the minimum of the objective function. We give a more exhaustive experimental result showing the performance degradation in terms of localization accuracy for different observer/target configurations in Section 8. The objective of the algorithm discussed in the following section is to handle this degradation in performance.

6 THE POLYHEDRA-BASED LOCALIZATION ALGORITHM: ALTERNATING PROJECTION APPROACH

The alternating projection algorithm is a method for finding a point in the intersection of multiple convex sets (e.g. polyhedra). It was initially proposed by John von Neumann [35] for the case when the convex sets are hyperplanes and then extended by Dykstra for the general case of any convex set [8]. Given convex sets, the algorithm is guaranteed to converge to a point that minimizes the distance to all convex sets. In this section, we show how to use the alternating projection approach to reduce the degradation in performance due to the geometric dilution of precision.

6.1 Localization Using the Alternating Projection Algorithm

Rethinking the polyhedra-based localization algorithm shown in the previous section, we can formulate the localization problem to that of finding a point that lies on the *boundary* of the intersection of all polyhedra generated by observers. Such an algorithm can be applied as follows. Starting from any arbitrary initial estimate, e.g. $\hat{z}_{\mathbb{T}}^{(0)} = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$, we start by projecting this estimate onto the boundary of the polyhedron $\mathcal{P}_1(A_1, b_{\mathbb{S}_1})$ generated by the first observer followed by a projection onto the boundary of the polyhedron $\mathcal{P}_2(A_2, b_{\mathbb{S}_2})$ and so on until we project onto all m polyhedra. By repeating the previous sequence of projections until we reach the maximum number of iterations, the alternating projection algorithm guarantees that the estimate converges to the target location. An example of the sequence of points generated by this algorithm is shown in Figure 5 for the two cases when (i) the target is inside

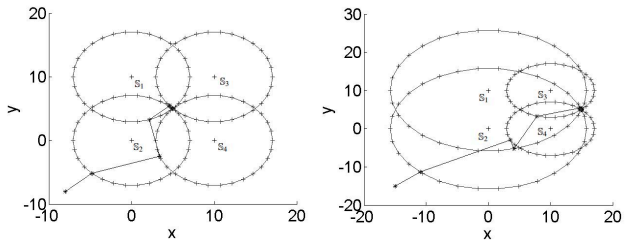


Figure 5: Several iterations of the alternating projection algorithm for different geometric configurations—a target inside (left) and outside (right) the observers' convex hull—showing the resilience of the polyhedra-based alternating projection to the geometric dilution of precision.

the convex hull of the observers and (ii) the target is outside the convex hull of the observers.

6.2 Projection Onto boundary of a Polyhedron

To complete the description of this algorithm, we need to show how the projection onto the boundary of a polyhedron can be computed. In this subsection, we describe how such a projection can be computed in plaintext. In the next subsection, we discuss how such an algorithm can be implemented in a privacy-preserving manner using additive homomorphic encryption.

The exact projection of a point $\hat{z}$ onto the boundary of a polyhedron $\mathcal{P}(A, b)$ can be computed using iterative methods that can be time consuming. To reduce this additional computation time, we rely on a light-weight heuristic for projection onto the boundary of each polyhedron which can be efficiently implemented later using homomorphic encryption while retaining good localization performance (as supported by the real-time experiments shown in Section 8). The proposed heuristic works as follows. Given a point $\hat{z}$ and a polyhedron $\mathcal{P}(A, b)$, we start by calculating the orthogonal distance between the point $\hat{z}$ and all the facets. This can be calculated as follows:

$$r_j = \frac{|a_j^T \hat{z} - b_j|}{\|a_j\|_2} \qquad a_j = A(j)^T, b_j = b(j), j \in \{1, \ldots, f\}$$

The next step is to compare the values of all the distances r_j to select the correct (minimum distance) facet j^*. Finally, we compute the projection of $\hat{z}$ onto the j^* facet as follows:

$$\hat{z}_{\Pi} = \begin{cases} \begin{bmatrix} 0 \\ \frac{b_{j^*}}{a_{j^*}(2)} \end{bmatrix} + \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - a_{j^*} a_{j^*}^T \right) \left(\hat{z} - \begin{bmatrix} 0 \\ \frac{b_{j^*}}{a_{j^*}(2)} \end{bmatrix} \right) & \text{if } a_{j^*}(2) \neq 0 \\ \begin{bmatrix} \frac{b_{j^*}}{a_{j^*}(1)} \\ 0 \end{bmatrix} + \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - a_{j^*} a_{j^*}^T \right) \left(\hat{z} - \begin{bmatrix} \frac{b_{j^*}}{a_{j^*}(1)} \\ 0 \end{bmatrix} \right) & \text{otherwise} \end{cases} \tag{6}$$

6.3 Poly-AP: Polyhedra-Based Alternating Projection Protocol

The `Poly-AP.Encrypt` and `Poly-AP.Decrypt` algorithms are similar to the Poly-LSQ protocol. Hence, we focus here on the `Poly-AP.Aggregate` algorithm as follows.

The protocol starts by query node $\mathbb{Q}$ generating a uniformly random initial estimate $\hat{z}_{\mathbb{T}}^{(0)}$, encrypting this estimate using its public key, and then sending $[\![\hat{z}_{\mathbb{T}}^{(0)}]\!]_{\text{pk}_{\mathbb{Q}}}$ back to the aggregator $\mathbb{A}$. It is crucial for the privacy of the Poly-AP that the initial estimate is not known to the aggregator $\mathbb{A}$. Upon receiving all messages from the observers and the initial state from the query node, the aggregator performs a projection onto the boundary of the polyhedron of the first observer followed by the second observer and so forth until reaching the maximum number of iterations.

The basic block of the alternating projection algorithm is to compute the distance between $[\![\hat{z}]\!]_{\text{pk}_{\mathbb{Q}}}$ and all the facets of a given polyhedron $\mathcal{P}(A, [\![b]\!]_{\text{pk}_{\mathbb{Q}}})$ and then to perform a comparison between these distances. Thanks to the polyhedron-based representation chosen in the previous section, we can use additive homomorphism to compute the encrypted version of the distances as follows. First note that the A matrix is revealed to all parties in plain text. Hence, the observer computes the factor $\frac{1}{\|a_j\|_2}$ in plain text followed by:

$$[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}} = \frac{1}{\|a_j\|_2} \otimes (a_j^T \otimes [\![\hat{z}]\!]_{\text{pk}_{\mathbb{Q}}} \ominus [\![b_j]\!]_{\text{pk}_{\mathbb{Q}}}) \tag{7}$$

To compute the absolute value $[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}$ we need to check the sign of $[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}$ (by comparing the value of $[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}$ to zero) and multiplying by -1 accordingly. Unfortunately, additive homomorphic encryption does not preserve order, so such comparisons cannot be directly computed. Therefore we rely on the the *secure comparison* protocol (introduced in Section 4.3) to perform comparison between $[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}$ and $[\![0]\!]_{\text{pk}_{\mathbb{Q}}}$ as follows.

Absolute Value Computation Protocol: Based on a random coin flip, the aggregator issues a call to the query node using either a SecCompare($[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}, [\![0]\!]_{\text{pk}_{\mathbb{Q}}}$) or a SecCompare($[\![0]\!]_{\text{pk}_{\mathbb{Q}}}, [\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}$) followed by sending the following encrypted values $[\![[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}$ and $[\![[\![0]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}$ using the same order used in the call of SecCompare. Once the query node computes the index of the argument with the maximum value, instead of returning this index it assigns the variables *maxArg* and *minArg* to $[\![[\![r_j]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}$ and $[\![[\![0]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}}$ according to its knowledge of which argument has the maximum value. Finally, the query node returns $[\![[\![|r_j|]\!]_{\text{pk}_{\mathbb{Q}}}]\!]_{\text{pk}_{\mathbb{A}}} = maxArg \ominus minArg$ to the aggregator. Once the aggregator retrieves the absolute value from the query node, it decrypts it using its own private key and the rest of the alternating projection algorithm follows accordingly. The correctness of the *absolute value computation protocol* is trivial and follows from the fact that one of the arguments is zero and hence does not affect the value of the addition.

The next step is to compare the values of $[\![|r_j|]\!]_{\text{pk}_{\mathbb{Q}}}$ for all the facets $j \in \{1, \ldots, f\}$. Similarly, this can be done by relying on the *secure comparison* protocol. Once the correct facet j^* is known, the final step is to compute the projection itself. Again thanks to the polyhedron representation and the fact that the A matrix does not need to be encrypted, the projection $[\![\hat{z}_{\Pi}]\!]_{\text{pk}_{\mathbb{Q}}}$ can be performed by applying additive homomorphism to (6). The privacy of the proposed algorithm can be summarized as follows.

Theorem 6.1. *Assume that all entities are honest-but-curious and under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the Poly-AP localization algorithm ensures (i) observer obliviousness, (ii) non-colluding aggregator obliviousness, and (iii) query node obliviousness.*

7 RESILIENT PRIVACY-AWARE LOCALIZATION

In previous sections, we focused primarily on one aspect of the problem, namely *privacy-preserving* localization. In this section, we demonstrate how the previously proposed algorithms can be augmented to achieve resilience as well. We base our resilience scheme on prior work on residue checking over noisy measurements [29], that was developed for the general case of secure estimation from heterogeneous sensors under false data injection attacks. In our scheme, location error (or residue) is computed using measurements from subsets of observers and compared against the maximum allowable residue. The subset of observers with the minimum residue is selected. A sufficient and necessary condition for this residue checking scheme to work is that the maximum number of malicious observers is $\lfloor \frac{m-3}{2} \rfloor$ [29]. While performing residue checking over subsets of observers is a combinatorial process, it was shown that using combinatorial search methods like satisfiability (SAT) solvers in conjunction with estimation methods leads to significant reduction in search time [29]. Indeed, such residue checking scheme does not work unless the localization error is bounded. Thanks to the polyhedra based localization algorithms discussed previously, and unlike differential privacy and other data corruption based privacy techniques, this condition comes for free from the construction of the protocol.

To augment the polyhedra based localization algorithms with the residue checking scheme, the aggregator follows the same steps discussed before to compute the location estimate $[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}}$ from all different subsets of $m - \lfloor \frac{m-3}{2} \rfloor$ observers. The next step is to compute the associated residual $[\![\xi]\!]_{\text{pk}_{\mathbb{Q}}} = \|\overline{A}[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}} - [\![\overline{b}_{\mathbb{S}}]\!]_{\text{pk}_{\mathbb{Q}}}\|$ where $\|.\|$ denotes the sum of absolute values. Thanks to polyhedra based representation, this residue can be computed using additive homomorphic encryption along with the absolute value protocol discussed before. Once all the residuals from all subsets of observers are calculated, the next step is to compare them to choose the minimum residue. This can be done by invoking the secure comparison protocol SecCompare($[\![\xi_1]\!]_{\text{pk}_{\mathbb{Q}}}, \ldots, [\![\xi_h]\!]_{\text{pk}_{\mathbb{Q}}}$) where h is the number of observer subsets. Finally, the location estimate $[\![\hat{z}_{\mathbb{T}}]\!]_{\text{pk}_{\mathbb{Q}}}$ corresponding to the subset of observers that produce the smallest residue is then sent to the query node for decryption. We call these algorithms resilient Poly-LSQ and resilient Poly-AP, respectively.

It is essential to note that by the end of the protocol, the query node learns only the location estimates produced by only one subset of observers (not all location estimates from all subsets of observers). Then, following the same algebraic construction shown in the proof of Theorem 5.1, we conclude that the resilience scheme described above does not leak any additional information and we conclude with the following result.

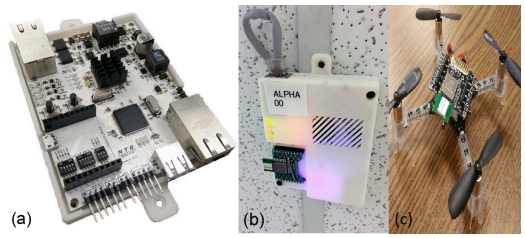


Figure 6: (a) Custom ranging anchor circuit board, (b) ceiling-mounted anchor node, and (c) mobile ranging target.

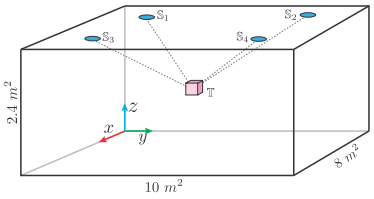


Figure 7: Ranging test bed configuration with four anchor nodes and one target node.

Theorem 7.1. *Assume that the number of malicious observers does not exceed* $\lfloor \frac{m-3}{2} \rfloor$ *and under standard cryptographic assumptions (namely the Decisional Composite Residuosity (DCRA) assumption), the resilient Poly-LSQ and resilient Poly-AP localization protocol produce resilient location estimates of the target while satisfying the same privacy guarantees in Theorem 5.1 and Theorem 6.1, respectively.*

8 RESULTS

In this section, we report the experimental results for a full, end-to-end analysis using the proposed protocols. We used the Paillier cryptosystem and the Java BigInteger library with full floating point support. Both the Aggregator and Querier are implemented on Macbook Pro laptops with Intel(R) Core i7-4870HQ CPUs @2.5GHz. We validated our proposed secure localization method using custom ranging hardware in an $8 \times 10 \times 2.4\ m^2$ room with four anchor (observer) nodes capable of symmetric double-sided time of flight range measurements and one mobile (target) node of unknown location as illustrated in Figure 7. The target node was constrained to motion on a fixed z plane, allowing the problem to be reduced to localization in $\mathbb{R}^2$. All experiments are performed while both fixed and moving obstacles are present in the room.

Anchor Ranging Hardware: The anchor nodes used in the following experiments consist of custom-built circuit boards equipped with ARM Cortex M4 processors at 196 MHz communicating to Decawave DW1000 ultra-wideband radios as shown in Figure 6. Each anchor node listens for incoming range messages from a target node and, upon reception, begins a symmetric double-sided ranging sequence. After completing a range sequence, each anchor communicates the range estimate to an Android Nexus 5 smartphone to perform the secure communication to the Aggregator localization server. In effect, the Nexus 5 and custom ranging hardware serve together as a single cohesive observer node.

Mobile Target Ranging Hardware Battery-powered mobile nodes also with ARM Cortex M4 processors based on the CrazyFlie 2.0 helicopter [1] and equipped with the very same DW1000 radio. This allows for compatibility in the double-sided ranging technique used.

It is important to note that the transmit power of the anchors is not known by external parties *a priori* so that the target nodes or any other eavesdropping node cannot determine precisely where the anchor nodes are. Furthermore, the anchor nodes are capable of changing their transmit power dynamically without detrimental effects on ranging estimates, further obfuscating their location [3].

8.1 Experiment 1: Execution Time

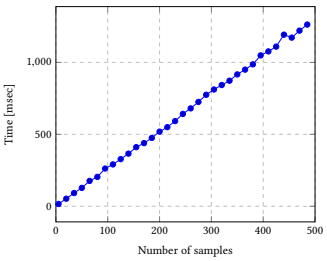
Execution time at Observers: At the observer side, execution is dominated by preparing the A matrix and encrypting the b vector. The execution time on the observers is shown in Figure 8(a), as a function of range circle samples. For 80 samples, as used in our experiments, execution time on the observers is below 250 ms.

Execution time at Aggregator: The number of alternating projection iterations has a large effect on the overall execution time at the aggregator side. Figure 8(b) shows that, while Poly-LSQ has a constant execution time of around 100 ms on the aggregator, Poly-AP has a linear dependency on the number of iterations, with 4 iterations requiring around 1200 ms on the aggregator. Finally, an increased number of samples of the ranging circle also causes an increased execution time on the Aggregator, with Poly-LSQ showing an execution time of under 100 ms for 80 samples while Poly-AP shows an execution time of around 1800 ms for 80 samples. While the Poly-AP algorithm requires considerably more computation than Poly-LSQ, it can still be executed in real time on commodity hardware.

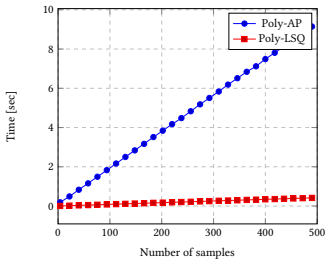
Next, we compare the execution time of the proposed protocols against two secure function evaluation schemes, namely (1) Fully Homomorphic Encryption [16] and (2) Garbled Circuits (implemented using the TASTY tool [20]). As shown in Figure 8(c), thanks to the polyhedra representation both the proposed Poly-LSQ and Poly-AP outperform the other algorithms by at least an order of magnitude. Moreover, and as expected, the FHE implementation is far away from being practical as it requires more than 4.3 hours to calculate the estimate of the target when measurements are available from only 3 observers. The Garbled Circuits scales better compared to FHE, however even for a small number of observers it requires multiple minutes of computation (on both observers and aggregator) to compute the target position. On the other hand, the execution time of the proposed Poly-LSQ is in the range of hundreds of milliseconds. These results show how our proposed protocols are better suited to real-time implementations of privacy-aware localization systems.

8.2 Experiment 2: Localization error

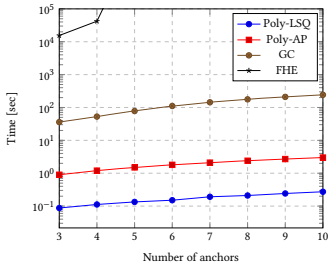
Effect of Number of Polyhedra Facets: We start by studying the effect of increasing the number of facets used to represent the polyhedra for each observer. Unlike Poly-LSQ, the alternating projection algorithm benefits greatly from increased polyhedra



(a) Execution time on observers vs. range circle samples.



(b) Aggregator execution time vs. range circle samples.



(c) Aggregator execution time (semi-log scale) vs. number of anchors.

Figure 8: Execution time Analysis

facets (circle samples). This is shown in Figure 9(a) where Poly-AP has converged to very high accuracy localization results after 10-20 polyhedra facets.

Resilience Against GDOP: We evaluated 40 test points in the 8×10 m room using each of the described localization protocols. Each range 'circle' is sampled 80 times to make the polyhedra used in Poly-LSQ and Poly-AP, and the number of iterations for the Poly-AP protocol is fixed at 4. Figure 9(b) shows the localization errors at each test point, where point IDs $\{1, \ldots, 14\}$ are within the polytope (convex-hull) formed by the edges connecting each anchor, and point IDs $\{15, \ldots, 40\}$ are outside. The localization error is calculated as the mean squared error between the location estimate and the ground truth location. As seen in Figure 9(b), The Poly-LSQ protocol shows the highest error, with excessive errors when the target is outside the anchors' convexhull due to the GDOP which is described in Subsection 5.3. The Poly-AP protocol provides good localization results compared to the traditional (unsecured) localization algorithm. Table 1 summarizes the average and standard deviation of the localization errors for the three protocols. Most importantly, the Poly-AP algorithm closely matches the performance of the state-of-art, unsecured localization algorithm. This lends credence to partial homomorphism as a viable method for privatizing observer data when performing high accuracy localization.

Table 1: Localization error comparison (m).

	mean	standard deviation
Unsecured Localization	0.2341	0.18738
Poly-AP	0.2381	0.18634
Poly-LSQ	1.1309	0.80183

8.3 Experiment 3: Communication Overhead

The localization methods described in this paper (including the unsecured least squares method) require some amount of communication between all parties involved—observers, aggregator, and the querying party. As discussed in Section 2.4, communication cost at the observer side is of great importance from a practical point of view. Every observer $\mathbb{S}$ in both the Poly-LSQ and Poly-AP protocols as well as the FHE implementation sends only one message to the Aggregator $\mathbb{A}$, with the difference that Poly-LSQ and Poly-AP send the information of all facets (80 in this experiment) while FHE sends only the information about the tuple (x_i, y_i, d_i). This eliminates the necessity of the observers to be online for longer times. On the other hand, GC requires multiple rounds of communication between the aggregator and all the observers, leading to an increase in bits communicated of two orders of magnitude. This is described in detail in Figure 8 (c).

8.4 Experiment 4: Resilience Against Data Injection Attacks

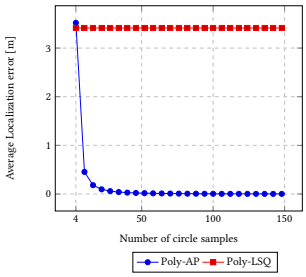
Finally, we study the performance of the resilient polyhedra based localization against malicious observers which launch a false data injection attacks. As expected from Section 7, and due to the combinatorial aspect, the execution time of the resilient scheme increases by a combinatorial factor of $\binom{m}{m-k}$ where $k = \lfloor \frac{m-3}{2} \rfloor$. The localization accuracy matches the ones shown in Experiment 2 since the resilience scheme is nothing more than multiple invocations of the localization protocols in Experiment 2.

9 CONCLUSION

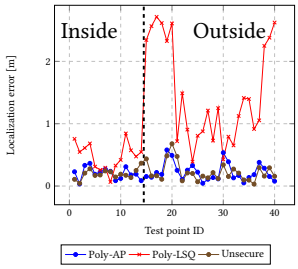
To conclude, we have presented PrOLoc, a set of protocols and algorithms designed to perform accurate localization in a manner that preserves the privacy of the observers whose measurements are used to derive the location estimate, while a subset of the same observers can be maliciously injecting false data. We have built PrOLoc around the Pallier additive homomorphic cryptosystem, redesigning traditional localization algorithms to benefit from the privacy guarantees of partial homomorphism. We provided results indicating that PrOLoc can yield location estimates that are comparable to state-of-art unsecured methods while operating in real time—more than 33,000× faster than a comparable fully homomorphic implementation and more than 500× faster than Garbled Circuits—while resulting in low communication cost at the observer side. In future, we plan to enhance the privacy guarantees for a more general setting where the aggregator node and the query node are allowed to collude.

ACKNOWLEDGMENTS

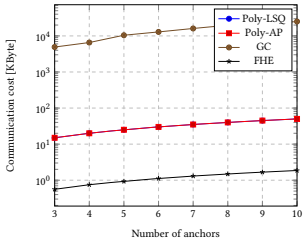
This research is funded in part by the National Science Foundation under awards # CNS-1329755 and ACI-1640813, and by the NIH Center of Excellence for Mobile Sensor Data to Knowledge under award # 1U54EB020404-01. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, NIH, or the U.S. Government.



(a) The effects of range circle 'samples' on localization error.



(b) Localization error of Poly-LSQ and Poly-AP protocols. The dotted line separates the points inside and outside the observers' convex hull.



(c) Communication cost (semi-log scale) vs Number of Anchors.

Figure 9: Localization error performance (left, center) and Communication cost (right).

REFERENCES

[1] Bitcraze CrazyFlie 2.0. https://www.bitcraze.io/. Accessed: 2017-01-01.

[2] Credit card data breach could affect 650,000 customers of J.C. Penny. http://www.nbcnews.com/id/22718442/ns/technology_and-science_security/t/credit-card-data-breach-could-affect/. Accessed: 2017-10-01.

[3] DecaWave DW1000 IR-UWB. http://www.decawave.com/products/dw1000.

[4] Slashdot. WordPress hacked, attackers get root access. http://it.slashdot.org/story/11/04/13/1925244/wordpress-hacked-attackers-get-root-access. Accessed: 2017-10-01.

[5] Dan Bogdanov, Riivo Talviste, and Jan Willemson. 2012. Deploying secure multiparty computation for financial data analysis. In *International Conference on Financial Cryptography and Data Security*. Springer, 57–64.

[6] Nicolás E. Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2014. Optimal Geo-Indistinguishable Mechanisms for Location Privacy. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS '14)*. 251–262.

[7] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. 2015. Machine Learning Classification over Encrypted Data. In *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2014*.

[8] JamesP. Boyle and RichardL. Dykstra. 1986. A Method for Finding Projections onto the Intersection of Convex Sets in Hilbert Spaces. In *Advances in Order Restricted Statistical Inference*, Richard Dykstra, Tim Robertson, and FarrollT. Wright (Eds.). Lecture Notes in Statistics, Vol. 37. Springer New York, 28–47.

[9] Srdjan Čapkun, Saurabh Ganeriwal, Farooq Anjum, and Mani Srivastava. 2006. Secure RSS-based localization in sensor networks. (2006).

[10] John R Douceur. 2002. The sybil attack. In *International Workshop on Peer-to-Peer Systems*. Springer, 251–260.

[11] Léo Ducas and Daniele Micciancio. 2015. FHEW: Bootstrapping homomorphic encryption in less than a second. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 617–640.

[12] Cynthia Dwork. 2006. Differential Privacy. In *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006) (Lecture Notes in Computer Science)*, Vol. 4052. Springer Verlag, 1–12.

[13] Cynthia Dwork. 2008. Differential Privacy: A Survey of Results. In *Proceedings of the 5th International Conference on Theory and Applications of Models of Computation (TAMC'08)*. 1–19.

[14] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy via Distributed Noise Generation. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques (EUROCRYPT'06)*. 486–503.

[15] Ravi Garg, Avinash L Varna, and Min Wu. 2010. Gradient descent approach for secure localization in resource constrained wireless sensor networks. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 1854–1857.

[16] Craig Gentry. 2009. *A Fully Homomorphic Encryption Scheme*. Ph.D. Dissertation. Stanford, CA, USA. Advisor(s) Boneh, Dan.

[17] Craig Gentry and Shai Halevi. 2011. Implementing Gentry's Fully-homomorphic Encryption Scheme. In *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology (EUROCRYPT'11)*. 129–148.

[18] Oded Goldreich. 2004. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press.

[19] Slawomir Goryczka, Li Xiong, and Vaidy Sunderam. 2013. Secure Multiparty Aggregation with Differential Privacy: A Comparative Study. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops (EDBT '13)*. 155–163.

[20] Wilko Henecka, Ahmad-Reza Sadeghi, Thomas Schneider, Immo Wehrenberg, and others. 2010. TASTY: tool for automating secure two-party computations. In *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 451–462.

[21] Yih-Chun Hu, Adrian Perrig, and David B Johnson. 2002. Wormhole detection in wireless ad hoc networks. *Department of Computer Science, Rice University, Tech. Rep. TR01-384* (2002).

[22] Yan Huang, Chih-hao Shen, David Evans, Jonathan Katz, and Abhi Shelat. 2011. Efficient Secure Computation with Garbled Circuits. In *Information Systems Security*, Sushil Jajodia and Chandan Mazumdar (Eds.). Lecture Notes in Computer Science, Vol. 7093. Springer Berlin Heidelberg, 28–48.

[23] S. U. Hussain and F. Koushanfar. 2016. Privacy preserving localization for smart automotive systems. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. DOI: http://dx.doi.org/10.1145/2897937.2898071

[24] Richard B Langley. 1999. Dilution of precision. *GPS world* 10, 5 (1999), 52–59.

[25] Kristin E. Lauter, Adriana López-Alt, and Michael Naehrig. 2015. Private Computation on Encrypted Genomic Data. *IACR Cryptology ePrint Archive* 2015 (2015), 133.

[26] Loukas Lazos and Radha Poovendran. 2004. SeRLoc: Secure Range-independent Localization for Wireless Sensor Networks. In *Proceedings of the 3rd ACM Workshop on Wireless Security (WiSe '04)*. ACM, New York, NY, USA, 21–30. DOI: http://dx.doi.org/10.1145/1023646.1023650

[27] Yehuda Lindell and Benny Pinkas. 2009. Secure Multiparty Computation for Privacy-Preserving Data Mining. *Journal of Privacy and Confidentiality* 1, 1 (2009).

[28] Qi Mi, John A. Stankovic, and Radu Stoleru. 2010. Secure Walking GPS: A Secure Localization and Key Distribution Scheme for Wireless Sensor Networks. In *Proceedings of the Third ACM Conference on Wireless Network Security (WiSec '10)*. 163–168.

[29] S. Mishra, Y. Shoukry, N. Karamchandani, S. Diggavi, and P. Tabuada. 2016. Secure State Estimation against Sensor Attacks in the Presence of Noise. *IEEE Transactions on Control of Network Systems* PP, 99 (2016), 1–1. DOI: http://dx.doi.org/10.1109/TCNS.2016.2606880

[30] Pascal Paillier. 1999. Public-key Cryptosystems Based on Composite Degree Residuosity Classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT'99)*. 223–238.

[31] Raluca Ada Popa, Hari Balakrishnan, and Andrew J. Blumberg. 2009. VPriv: Protecting Privacy in Location-based Vehicular Services. In *Proceedings of the 18th Conference on USENIX Security Symposium (SSYM'09)*. USENIX Association, Berkeley, CA, USA, 335–350.

[32] Elaine Shi, Richard Chow, T h. Hubert Chan, Dawn Song, and Eleanor Rieffel. 2011. Privacy-preserving aggregation of time-series data. In *Proceedings of the Network and Distributed System Security Symposium (NDSS 2011)*.

[33] Tao Shu, Yingying Chen, Jie Yang, and A. Williams. 2014. Multi-lateral privacy-preserving localization in pervasive environments. In *INFOCOM, 2014 Proceedings IEEE*. 2319–2327.

[34] Nils Ole Tippenhauer and Srdjan Capkun. 2008. UWB-based secure ranging and localization. *month* 1, 586 (2008), 12.

[35] J. Von Neumann. 1950. *Functional Operators: The Geometry of Orthogonal Spaces*. Princeton University Press.

[36] Martin J Wainwright, Michael I Jordan, and John C Duchi. 2012. Privacy aware learning. In *Advances in Neural Information Processing Systems*. 1430–1438.

[37] Wei Wang, Yin Hu, Lianmu Chen, Xinming Huang, and B. Sunar. 2015. Exploring the Feasibility of Fully Homomorphic Encryption. *Computers, IEEE Transactions on* 64, 3 (2015), 698–706.

[38] Andrew C. Yao. 1982. Protocols for Secure Computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS '82)*. 160–164.