Privacy Preserving Localization for Smart Automotive Systems

Siam U. Hussain, Farinaz Koushanfar
Department of ECE
University of California, San Diego, CA 92093
{siamumar, fkoushanfar}@ucsd.edu

ABSTRACT

This paper presents the first provably secure localization method for smart automotive systems. Using this method, a lost car can compute its location with assistance from three nearby cars while the locations of all the participating cars including the lost car remain private. This localization application is one of the very first location-based services that does not sacrifice accuracy to maintain privacy. The secure location is computed using a protocol utilizing Yao's Garbled Circuit (GC) that allows two parties to jointly compute a function on their private inputs. We design and optimize GC netlists of the functions required for computation of location by leveraging conventional logic synthesis tools. Proof-of-concept implementation of the protocol shows that the complete operation can be performed within only 550 ms. The fast computing time enables practical localization of moving cars.

Keywords

Connected Cars, Secure Automotive System, Location Privacy, Location Based Services, Secure Function Evaluation, Garbled Circuit

1. INTRODUCTION

Contemporary automobiles are increasingly being equipped with advanced technologies that make significant enhancements to both functionality and safety of the vehicles. Two of the most significant improvement in this field are smart navigation system and inter-vehicle communications. Each modern vehicle also includes an intra-network of processors connected to a central CPU providing Ethernet, USB, Bluetooth, and IEEE 802.11 interfaces [1]. Besides enhancing performance, these technologies also create new dimensions for attack. Thus, in addition to classic vehicular reliability requirement, security and privacy of the user should be taken into careful consideration while implanting these advanced features [1–3]. Moreover, due to the increasing reliance on these smart features, backup plans to cope with the failure of one or more components is also crucial for reliability.

In this paper, we present the first private localization method for smart cars based on provably secure primitives. With this method, a car, lost due to unavailability of GPS, can send requests to three nearby cars to get assistance in finding its location. The three assisting cars then engage in a privacy preserving triangle localization protocol to estimate the location of the lost car. The locations of all the cars including the lost car remain private.

To date, the most widely explored method to ensure user privacy in Location Based Services (LBS) is location cloaking [4–6]. In this method, instead of sending the exact location and time instant of the user, a range of area covered in a period of time is sent. To make sure that user's location cannot be inferred from this data, the range and period is chosen such that there are at least $k-1$ other users in that area during that period, which ensures "k-anonymity" of the user. k-anonymity requires the existence of a trusted third party called anonymizer that combines the user location with locations of other users subscribed to the service. This anonymizer presents a single point to attack the system. Moreover, cloaking is also vulnerable to context based attack and trajectory-tracing. More importantly, the approximate location results in noisy and stochastic response to the query. While this approximate response may be acceptable in some LBS scenario, for localization and navigation applications the accuracy of the method is crucial.

The work in [7–9] explored performing the location-based query (e.g., nearest neighbor) in a transformed space. While these methods increase the accuracy over the cloaking approaches, they still have few drawbacks. For example, [7] propose three methods that either requires a semi-trusted third party or has to sacrifice accuracy or privacy for simplified operation. The authors in [7, 9] consider the privacy of only one party (client), while the data of the other party (server) is assumed public.

To compute accurate results while maintaining complete privacy of all the participating parties, we employ Yao's Garbled Circuit (GC) protocol [10] for Secure Function Evaluation which is currently considered to be the most effective provable privacy-preserving technique [11, 12]. This protocol allows two parties to jointly evaluate a function on inputs which are encrypted to maintain privacy. Unlike the previous methods, this protocol does not involve trade-off between accuracy and privacy. To date, very few work have considered GC for LBS applications. Ours is the first privacy-preserving localization method based on GC.

DAC '16, June 05-09, 2016, Austin, TX, USA

DOI: http://dx.doi.org/10.1145/2897937.2898071

We devise a protocol where the three assisting cars participate in a total six invocations of the 2-party GC operation such that the locations of all cars including the lost car remain private. To cope with the time constraint due to car movement, the protocol is designed such that each car can simultaneously participate in two GC operations with each of the two other cars (assuming a multi-core architecture of the processors, which is widely available at present).

In GC, the pertinent function is represented as a list of Boolean logic gates, called *netlist*. We generate the netlists required for the localization protocol by using conventional logic synthesis tools with GC optimized custom libraries as suggested in [13]. Our custom synthesis library includes the first GC optimized implementations of division and square root functions, required for the computation of the location of the lost car. The synthesis library presented in [13] include implementations of unsigned addition, subtraction, and multiplication. We add enhanced implementations of these functions to our library to support signed inputs and overflow.

One major use case for our privacy-preserving localization is in military applications when a lost military vehicle requires help in locating itself. It is crucial that the location of each participating vehicle remain private so that an adversarial vehicle cannot learn their location by pretending to be an ally or by tapping into the common channel. This application can also be beneficial in verifying a suspected vehicles claimed location via distance bounding with assist from three nearby cars. Generally, three verifying base stations perform distance bounding on the suspected vehicle confining it to a triangular region. However, this requires costly infrastructure which may not be available in all places. In this scenario, three other cars can act as the verifying base stations while their locations remain private.

Contributions: In brief, our contributions are as follows:

- We present the first provably secure triangle localization for smart automotive systems. We design a protocol utilizing 2-party GC operation such that a lost car along with three nearby cars can jointly compute the location of the lost car while the locations of all the participating cars remain private.
- We develop a circuit synthesis library with functions required to generate GC optimized netlists for triangle localization algorithm. This library includes the first ever GC implementations of square-root and division operations.
- Proof-of-concept implementation of our protocol demonstrates practicality of the design. The complete protocol is performed within only 550 ms.

2. GLOBAL FLOW

The overview of the localization process is displayed in Fig. 1. The lost car Q sends requests to three nearby cars A, B, and C to assist in computing its location. Each assisting car X estimates its distance r_X from Q. Then A, B, and C participate in a privacy preserving localization protocol based on Yao's GC operation to compute the location of Q. The inputs from each assisting car X are its location L_X and its distance r_X from Q. Ideally, the location of Q would be a common intersection of three circles centered at A, B and C. However, due to inaccuracy in distance estimation, the location of Q is computed as the median of a triangle.

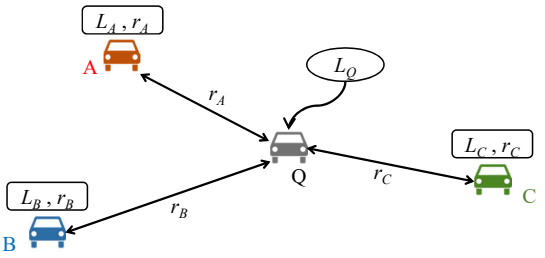

Figure 1: Overview of the Localization Algorithm

Each pair of cars (say A and B) participates in a GC operation to compute two possible candidates for one vertex of the triangle. Then one of them (say B) participates in another GC operation with the third car (C) to select the candidate closer to C as the vertex. Thus, six GC operations are required to determine all three vertices of the triangle. One car can learn zero to at most two vertices, therefore, a single car cannot compute the median on its own. Throughout the protocol, the input set $\{L_X, r_X\}$ of a car X is not revealed to any other participating car. The median L_Q, the location of Q, is computed through *secure sum* [14] protocol where all four cars participate and revealed only to Q.

3. PRELIMINARIES

3.1 Cryptographic Protocols

In this section, we provide a brief description of the background related to the current work. Consistent with most work in this area, we assume an *honest-but-curious* attack model[15, 16], where the participating parties follow the agreed upon protocol, but may want to deduce more from the information at hand. This can be readily modified to support *malicious* model by following the methodologies presented in [17].

Oblivious Transfer. Oblivious Transfer (OT)[18] is a cryptographic protocol executed between a sender S and a receiver R, where R selects one from a pair of messages provided by S without revealing her selection. In an 1-out-of-2 OT protocol, (OT_1^2), S holds a pair of messages (m_0, m_1); R holds a selection bit $b \in 0, 1$ and obtains m_b without revealing b to S and learns nothing about m_{1-b}.

Garbled Circuit. Yao's Garbled Circuit (GC)[10] is a cryptographic protocol where two parties Alice and Bob jointly compute a function $z = f(x, y)$ on their private inputs x, provided by Alice and y, provided by Bob. The function f is represented as a Boolean circuit, called *netlist*, consisting of 2-input 1-output logic gates. Alice, called the *garbler*, garbles the circuit as follows. She assigns each wire in the netlist with two k-bit random keys corresponding to the values 1 and 0. For each gate, a garbled truth table is generated by encrypting the keys for output with corresponding input keys. She then sends the garbled circuit along with the keys corresponding to her input values to Bob, called the *evaluator*. Bob obtains his keys corresponding to his input values obliviously through 1-out-of-2 OT protocol. He then uses these input keys to evaluate the encrypted tables gate by gate. Finally, Alice and Bob share their output maps, which can be configured to let one or both of them learn the output z.

A number of optimizations to the GC protocol have been

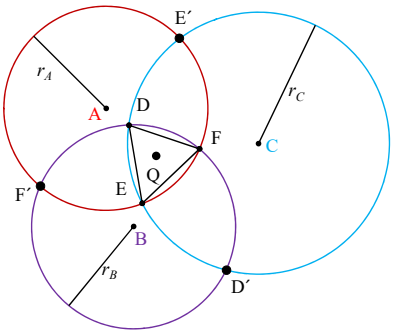

Figure 2: Triangle Localization Algorithm. The lost car is Q and the assisting cars are A, B, and C. The calculated location of Q is the centroid of the ΔDEF.

proposed: free-XOR[19], row reduction[20], half gate[21], and fixed-key cipher[16]. Among these optimizations, the a major one is free-XOR as it allows the evaluation of XOR, XNOR and NOT gates without costly cryptographic encryption, which also translates to less communication time as the XOR gates does not need transfer of the garbled tables. As a result, the primary optimization goal while generating the netlist for f is to minimize the number of non-XOR gates. According to the methodology presented in [13], f is first described with a Hardware Description Language (HDL) and compiled with a logic synthesis tool using libraries that are designed to minimize the number of non-XOR gates.

3.2 Triangle Localization

Fig. 2 shows the setup of the triangle localization algorithm with the lost car as Q and the assisting cars as A, B, and C. First, distances r_A, r_B, and r_C of Q from A, B, and C respectively, are estimated. If distances could be estimated accurately, the three circles centered at A, B, and C with radii r_A, r_B, and r_C, respectively, would have a common intersection at Q. However, in practice they cannot be estimated so precisely. Therefore, they are typically overestimated so that a triangle DEF is formed by the points of intersections. The location of Q is the median of ΔDEF. Two circles may have two intersections. The one that falls inside the third circle forms one vertex of the triangle.

The equations for calculating the coordinates of F and F' is provided here [22]. Equations (1) and (2) have two solutions as given by (4) and (5). The one inside the range of C, decided through inequality (3), forms one vertex of the triangle. The vertex of the triangle is shown as F in the figure just for simplicity, it could be either of F or F'.

$$\sqrt{(x_F - x_A)^2 + (y_F - y_A)^2} = r_A \quad (1)$$

$$\sqrt{(x_F - x_B)^2 + (y_F - y_B)^2} = r_B \quad (2)$$

$$\sqrt{(x_F - x_C)^2 + (y_F - y_C)^2} \leqslant r_C \quad (3)$$

$$x_F = (1/2p)(y_F q + t) \quad (4)$$

$$y_F = \frac{1}{p^2+q^2}(pqx_A + y_B p^2 - \frac{1}{2}qt \pm \frac{1}{2}\sqrt{(qt - 2y_A p^2 - 2pqx_A)^2 - s(p^2+q^2)}) \quad (5)$$

$$where,\ p = x_B - x_A,\ q = y_B - y_A$$
$$t = r_A^2 - r_B^2 + x_B^2 - x_A^2 + y_B^2 - y_A^2$$
$$s = (4p^2y_A^2 + t^2 - 4ptx_A + 4p^2x_A^2 - 4p^2r_A^2)$$

4. PROTOCOL AND ANALYSIS

4.1 Protocol

There are two phases in the protocol to securely compute the location of the lost car. In the first phase, the coordinates of the triangle DEF are computed through six invocations of the GC protocol. For the location verification scenario, the coordinates are provided to the verifying authority after this phase. For other localization scenarios, the median of the triangle is computed through the *Secure Sum*[14] protocol in the second phase.

Phase 1: Computing triangle DEF

For this phase we need to evaluate the following two functions through GC. Similar to the previous section, the computation of the vertex F is used as an example here.

$[x_F, y_F, x'_F, y'_F] = Intersection(x_A, y_A, r_A, x_B, y_B, r_B)$, that implements Eq. (4) and (5).

$in_F = Range(x_F, y_F, x_C, y_C, r_C)$, that implements inequality (3).

The steps of this phase are as follows.

i A, B, and C estimate their distances r_A, r_B, r_C respectively with Q.

ii A and B compute the coordinates $F(x_F, y_F)$ and $F'(x'_F, y'_F)$ of the intersections of their circles by evaluating the $Intersection$ function though Yao's GC protocol. The output map is configured such that A learns $F(x_F, y_F)$ and B learns $F'(x'_F, y'_F)$.

iii B and C jointly decide whether F' lies inside the range of C by evaluating the *Range* function though Yao's GC protocol. The output in_F is 1 if F' lies inside the range of C, and 0 otherwise, in which case the intersection F lies inside the range of C. B learns in_F and shares it with A. C learns nothing in this step.

iv B and C perform the Step ii. B learns $D(x_D, y_D)$ and C learns $D'(x'_D, y'_D)$.

v C and A perform the Step iii to compute in_D which is 1 if D' lies inside the range of A or 0 if D lies inside the range of A. C learns in_D and shares it with B. A learns nothing in this step.

vi C and A perform the Step ii. C learns $E(x_E, y_E)$ and A learns $E'(x'_A, y'_A)$.

vii A and B perform the Step iii to compute in_E which is 1 if E' lies inside the range of B or 0 if E lies inside the range of B. A learns in_E and shares it with C. B learns nothing in this step.

Phase 2: Computing the median of ΔDEF

After phase 1, each assisting car possesses the coordinates of two intersections and two Boolean variables indicating whether or not these intersections are vertices of ΔDEF. In this phase, the assisting cars along with the lost car Q compute the median of the triangle through the following steps.

i Q sends a random coordinate (x, y) to A.

ii A computes the sums $X_A = (x + \overline{in}_F.x_F + in_E.x'_E)$ and $Y_A = (y + \overline{in}_F.y_F + in_E.y'_E)$ and sends to B.

iii B computes the sums $X_B = (X_A + \overline{in}_D.x_D + in_F.x'_F)$ and $Y_B = (Y_A + \overline{in}_D.y_D + in_F.y'_F)$ and sends to C.

iv C computes the sums $X_C = (X_B + \overline{in}_E.x_E + in_D.x'_D)$ and $Y_C = (Y_B + \overline{in}_E.y_E + in_D.y'_D)$ and sends to Q.

v Q now subtracts the initial random numbers from the sums and compute the medians as $((X_C - x)/3$, $(Y_C - y)/3)$ which are the coordinates of its location.

4.2 Distance Compensation

According to the protocol described in the previous section, one assisting car may know two vertices of ΔDEF. The estimated location of Q is the median of ΔDEF and is calculated through the secure sum protocol such that only Q learns the final result. However, if the area of the triangle is too small, the location of Q may be estimated by a car with good accuracy from just two vertices of ΔDEF. To prevent this, Q should be allowed to manipulate the area of ΔDEF by controlling the estimated distances from the three assisting cars. On the other hand, the estimated distance should only be known to the respective assisting car.

Among several methods for distance estimation, the one most suitable for this purpose is the two-way Time of Arrival method [23]. In this method, the assisting car sends a synchronization message to the lost car who sends it back. The distance is estimated from the message propagation time. To increase the estimated distance, the lost car can wait an arbitrary amount of time before returning the message. Note that since the final location is the median of the triangle, the larger area does not result in a significant error in the estimated location as we will show in Section 6.

4.3 Security Analysis

We now analyze what information each cars can learn regarding the location of the other cars.

Lost Car. In the protocol and the distance calculation method described in this section, the lost car learns nothing but its own location. However, there is a maximum range within which the cars will be able to communicate with each other. If that range is R, the lost car can assume that the three assisting cars are within a circular area around it with a radius of R. Therefore the uncertainty over the location of the assisting cars is $1/\pi R^2$.

Assisting Cars. An assisting car can be interested in two types of information: the locations of the other two assisting cars and the location of the lost car. Each assisting car knows only one of the intersections with the circle of the other two assisting cars. With this information, it is not possible to deduce the center of the other circle. Therefore, the uncertainty for one assisting car over the location of other two assisting cars is $1/\pi R^2$.

Considering the location of the lost car, an assisting car knows the distance between the lost car and itself with some uncertainty created by the lost car by modifying the propagation time. Therefore, an assisting car X (= A or B or C) can confine the location of the lost car within a circular region with radius r_X. It is possible for one assisting car to know the coordinates of two of the vertices of the ΔDEF. Those two vertices form one chord of that circle. In a strict sense, it is not possible to learn which side of that chord the other vertex resides. However, if the two partitions on either side of the chord have largely different areas, it is more likely that the lost car is on the larger partition. Even though it is not straight forward to calculate the uncertainty here, the minimum uncertainty in this case would be $2/\pi r_X^2$.

5. GC OPERATION

5.1 Netlist Generation

Intersection. To generate the GC optimized netlist for the *Intersection* function that computes Eq. (4) and (5) we need the implementations of arithmetic functions with the minimum number of non-XOR gates which minimizes both the number of communication and computation [19]. Our custom synthesis library includes the first GC optimized implementations of the division and square-root functions. Moreover, implementations of all the arithmetic functions support signed inputs with variable bit-length and overflow, which are essential for generating netlist for any arbitrary practical function.

In our implementation, the number of non-XOR gates in a W bit division operation is $\mathcal{O}(W^2)$ which is similar to the complexity of the multiplication operation provided in [13]. The number of non-XOR gates for a 64-bit division is 12,546.

The square root operation follows an iterative procedure. The number of non-XOR gates in a W bit square root operation with K iterations is $\mathcal{O}(W^2K)$. Again, the number of required iterations can be assumed to be linearly proportional to the bit width, which simplifies the term to $\mathcal{O}(W^3)$. The number of non-XOR gates for a 64-bit square root operation with 32 iterations is 12,733.

Range. Even though inequality (3) involves square-root operation, both sides of this inequality are positive quantities as both of them are measured distances. Therefore, we can avoid the costly square-root operation by squaring both sides. As a result, the netlist for this function is much smaller than the *Intersection* netlist.

The netlists for each function need to be generated only once. It is generated offline and saved in each car's memory.

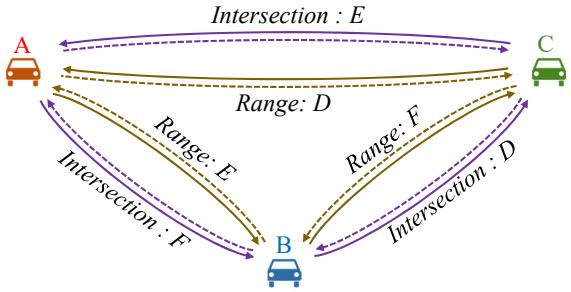

Figure 3: Illustration of parallel invocations of GC protocol.

5.2 Invocation of GC Protocol

Each of the assisting cars participates in two GC operations on the *Intersection* function with the other two cars. These two GC operations are independent of each other and performed in parallel in two cores of the processor. To ensure symmetry, each car performs as the garbler for one pair and the evaluator for the other. Similarly, each assisting car participates in two parallel GC operations on the *Range* function with the other two cars. Fig. 3 illustrates these operations. The outer arrows depict GC on *Intersection* and the inner arrows depict GC on *Range*. The vertex of

ΔDEF that is being computed in each GC operation is also indicated beside the arrows. A solid arrow emanating from a car indicates that the car acts as the garbler in that operation, and a dashed arrow indicates the evaluator.

The operation of the car A is described here as an example. A acts as the garbler while B acts as the evaluator to determine the coordinates of F and F' through the *Intersection* function and only learns the coordinate of F. In parallel to this, A participates in another GC operation as the evaluator, with C as the garbler to compute the coordinates of E and E' and learns only the coordinate of E'. A then performs as the garbler, while B performs as the evaluator to decide whether E' forms one vertex of the triangle through the *Range* function and shares the result with C. At the same time, it acts as the evaluator in another GC operation where C is the garbler to decide whether D' forms one vertex of the triangle without learning the result.

6. EVALUATION

6.1 Error Analysis

We first analyze the error in the location estimated by triangle localization algorithm. Note that this error is solely due to the localization method, and distance estimation error. The protocol does not introduce any additional error.

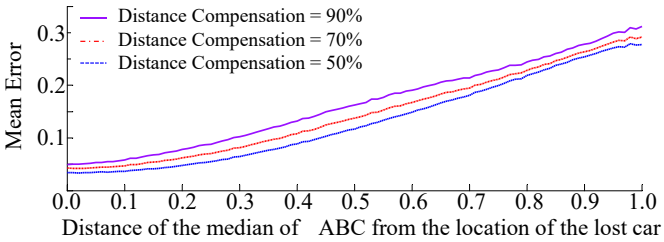

Figure 4: Normalized mean error in the estimated location of the lost car as a function of the normalized distance between the actual location of the lost car and the median of ΔABC with different degrees of distance compensation

To estimate the error, we run simulation by placing the assisting cars at random positions inside a square area with dimension T and place the lost car at the center of that square. The error is quantified as the Euclidean distance between the estimated and actual location of the lost car, normalized to T. Since the estimation error depends on the positions of the assisting cars with respect to the lost car, the error is plotted in Fig. 4 against the distance (normalized to T) between the actual location of the lost car and the median of the triangle formed by cars A, B, and C. For each point on the curves, the simulation is run for $5.7E+03$ times. To analyze the effect of distance compensation, we simulate three cases where the actual distance is increased by 50%, 70%, and 90%, respectively. The plot shows that the estimation errors are fairly close for all three cases.

6.2 Circuit Synthesis

Two netlists are required for the GC operations- *Intersection* and *Range*. The functions are described using Verilog HDL and compiled with Synopsys Design Compiler. The results of the synthesis is presented in Table 1. As already mentioned, the total time depends only on the number of non-XOR gates.

Table 1: Number of XOR and non-XOR gates in the netlists

Function	Intersection	Range
No of non-XOR gates	5.38E+04	7.83E+02
No. of XOR gates	1.74E+05	1.71E+03
Total	2.27E+05	2.49E+03

6.3 Timing

To assess the timing performance, we run the localization protocol on a system with Ubuntu 14.10 Desktop, 12.0 GB of memory, and Intel Core i7-2600 CPU @ 3.4GHz. The number of clock cycles to garble/evaluate each netlist once is presented in Table 2.

Table 2: Timing results

Function	Intersection		Range	
	Garbler	Evaluator	Garbler	Evaluator
OT	4.97E+08	5.54E+08	4.40E+08	4.43E+08
Communication	1.19E+06	3.42E+07	1.62E+05	2.73E+07
Garbling/Evaluation	3.28E+07	2.11E+07	3.45E+06	1.69E+07

Each netlist is garbled/evaluated 3 times by the three cars in parallel. The total number of clock cycles from the lost car initiating the operation to the final computation of its location is $1.89E+09$ which translates to only 550 ms.

Even though the evaluation is performed on a desktop PC, this protocol is practical with processors available in smart cars today. For example, Intel Atom Processor E3845, designed for in-vehicle solutions, has four cores operating at 1.91GHz and an L2 cache of 2MB [24]. The protocol requires transmission of about 1MB of data. With transmission speed in MHz range [25], the transmission time is within practical limits. The memory footprint of this operation is about 1.8MB, which can fit in the L2 cache of an Atom processor.

7. RELATED WORK

There are a number of works that designed privacy preserving Location Based Services (LBS) based on cryptographic primitives. Methods for privacy preserving nearest neighbor search are presented in [7, 9]. The work in [7] employs one-way Hilbert transformation to map the space of all elements to another space and resolve the query in that transformed space. It requires a trusted third party to perform the transformation in an offline phase. The method presented in [9] confines each point of interest (POI) to a cell, named a Voronoi cell, such that the POI is the nearest neighbor to any point that falls within that cell. Then a regular rectangular grid is super-imposed over this Voronoi diagram. A user retrieves all the Voronoi cells intersecting the region she belongs to on the grid through private information retrieval method and locally compute the nearest neighbor. Both these methods consider privacy of the query only, the database of the POIs is assumed to be public. Three methods based on homomorphic encryption to find if two friends are nearby without revealing their locations is presented in [8]. There are different trade-offs involved in these methods: they either require a semi-trusted third party or sacrifice accuracy or privacy for simplified operation.

The work in [26] presents application specific solutions based on GC to some problems in location-based services. They solve basic problems like point-inclusion (whether or not one party's point is included in other party's polygon),

intersection (whether or not two polygons from two users have an intersection), closest pair (form a pair closest of points taking one point from each set provided by two users). A GC based method to compute the nearest neighbor of a group of people is presented in [27]. In this method, two users participates in GC protocol to compute the nearest neighbor of the group. The other members of that group receive their input keys through OT from the garbler and share them with the evaluator. This creates a security threat as the collusion between only two users will reveal the location of all other members of the group. A scalable privacy preserving k-nearest neighbor search is presented in [28] which utilize sequential description of GC[13].

8. CONCLUSION

We present the first provably secure localization protocol that allows a lost car to compute its location with assistance from three nearby cars. The protocol employs the SFE technique named Yao's Garbled Circuit (GC) for the computations jointly performed by all the cars to determine the location of the lost car without revealing their own locations. Our localization method is one of the very first location-based services that does not involve any trade-off between accuracy and privacy. We design netlists for the functions required for computation of location and compiled them with conventional logic synthesis tool using custom libraries that incorporate implementations of arithmetic operations optimized for the GC. Our implementation demonstrates that the localization operation is completed within only 550 ms, a time period short enough to localize moving cars.

Acknowledgments. This work is supported in parts by an Office of Naval Research grant (ONR-R17460), National Science Foundation grants (CNS-1059416 and CCF-1116858), Semiconductor Research Corporation grant (2013-HJ-2471) and Multidisciplinary Research Program of the University Research Initiative grant (FA9550-14-1-0351).

9. REFERENCES

[1] J. Hubaux, S. Capkun, and J. Luo, "The security and privacy of smart vehicles," in *IEEE S & P*, 2004.

[2] "Automotive security best practices - intel," 2015.

[3] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J. Hubaux, "Secure vehicular communication systems: design and architecture," in *IEEE CM*, 2008.

[4] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar, "Preserving user location privacy in mobile data management infrastructures," in *Privacy Enhancing Technologies*, Springer, 2006.

[5] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," in *IEEE ITKDE*, 2007.

[6] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *ICMSAS*, ACM, 2003.

[7] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *ASTD*, Springer, 2007.

[8] G. Zhong, I. Goldberg, and U. Hengartner, "Louis, lester and pierre: Three protocols for location privacy," in *Privacy Enhancing Technologies*, Springer, 2007.

[9] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K. Tan, "Private queries in location based services: anonymizers are not necessary," in *SIGMOD ICMD*, ACM, 2008.

[10] A. Yao, "How to generate and exchange secrets," in *IEEE FOCS*, 1986.

[11] Y. Huang, D. Evans, and J. Katz, "Private set intersection: Are garbled circuits better than custom protocols?," in *NDSS*, 2012.

[12] Brenner, Perl, and Smith, "hcrypt SFE project." https://hcrypt.com/sfe/.

[13] E. M. Songhori, S. U. Hussain, A. Sadeghi, T. Schneider, and F. Koushanfar, "Tinygarble: Highly compressed and scalable sequential garbled circuits," in *IEEE S&P*, 2015.

[14] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu, "Tools for privacy preserving distributed data mining," in *SIGKDD Explorations Newsletter*, 2002.

[15] B. Kreuter, A. Shelat, B. Mood, and K. R. Butler, "PCF: A portable circuit format for scalable two-party secure computation.," in *USENIX Security*, 2013.

[16] M. Bellare, V. T. Hoang, S. K., and P. Rogaway, "Efficient garbling from a fixed-key blockcipher," in *IEEE S&P*, 2013.

[17] Y. Lindell and B. Pinkas, "Secure two-party computation via cut-and-choose oblivious transfer," in *Journal of Cryptology*, Springer, 2012.

[18] M. Naor and B. Pinkas, "Computationally secure oblivious transfer," in *Journal of Cryptology*, Springer, 2005.

[19] V. Kolesnikov and T. Schneider, "Improved garbled circuit: Free xor gates and applications," in *ICALP*, Springer, 2008.

[20] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," in *CEC*, ACM, 1999.

[21] S. Zahur, M. Rosulek, and D. Evans, "Two halves make a whole: Reducing data transfer in garbled circuits using half gates." Cryptology ePrint Archive, 2014. http://eprint.iacr.org/2014/756.

[22] Y. Shang, Z. Liu, J. Wang, and X. Xiao, "Triangle and centroid localization algorithm based on distance compensation," in *ICISCE*, IET, 2012.

[23] A. Bensky, *Wireless positioning technologies and applications*. Artech House, 2007.

[24] "Intel Atom Processor E3845." ark.intel.com/products/78475, 2015.

[25] "IEEE 1609 - family of standards for wireless access in vehicular environments (WAVE)." standards.its.dot.gov/factsheets/factsheet/80, 2009.

[26] M. Atallah and W. Du, "Secure multi-party computational geometry," in *Algorithms and Data Structures*, Springer, 2001.

[27] Y. Huang and R. Vishwanathan, "Privacy preserving group nearest neighbour queries in location-based services using cryptographic techniques," in *IEEE GLOBECOM*, 2010.

[28] E. Songhori, S. Hussain, A. Sadeghi, and F. Koushanfar, "Compacting privacy-preserving k-nearest neighbor search using logic synthesis," in *DAC*, 2015.