Delphine Reinhardt, Ilya Manyugin: *OP4: An OPPortunistic Privacy-Preserving Scheme for Crowdsensing Applications*. In: Proceedings of the 41st IEEE International Conference on Local Computer Networks (LCN), pp. 460-468, November 2016.

OP^4: An OPPortunistic Privacy-Preserving Scheme for Crowdsensing Applications

Delphine Reinhardt*†, Ilya Manyugin*
* Rheinische Friedrich-Wilhelms-Universität, Bonn, Germany
† Fraunhofer FKIE, Wachtberg, Germany
Emails: delphine.reinhardt@cs.uni-bonn.de, ilya.manyugin@gmail.com

***Abstract*—Crowdsensing applications rely on volunteers to collect sensor readings using their mobile devices. Since the collected sensor readings are annotated with spatiotemporal information, the volunteers' privacy may be endangered. Existing privacy-preserving solutions often disclose the volunteers' location information to either a central third party or their peers. As a result, the volunteers need to trust these parties to respect their privacy. In this paper, we present a distributed approach based on the concept of multi-party computation, which does not require a trusted party and protects the location information against curious users. We evaluate the performance of our approach and show its feasibility by means of extensive simulations based on a real-world dataset. We further implement a proof-of-concept to test its performance under realistic conditions.**

I. Introduction

Current mobile devices, such as smartphones and tablets, are ubiquitous and offer an increasing number of resources and embedded sensors. The combination of these aspects has led to the emergence of multiple sensing paradigms, such as *participatory sensing* [2] and *mobile crowdsensing* [11]. These paradigms share the same underlying concept leveraging volunteers to collect sensor readings with their personal devices. Without loss of generality, we therefore use the generic term of *crowdsensing* to designate such applications in the remainder of this paper. Based on this concept, a myriad of applications have already been proposed, ranging from documenting participants' health conditions to monitoring their surrounding environment [9].

A challenge in these applications is the protection of the participants' privacy because the participants report the collected sensor readings along with spatiotemporal information to the application server [9]. Crowdsensing administrators hence gain access to the participants' whereabouts during the collection campaign. This information can lead to the inference of additional sensitive insights about the participants, such as their medical state based on frequent visits to hospitals or political views when attending political meetings [35].

Different mechanisms have been presented to protect the participants' location privacy. These mechanisms however require the participants to reveal their locations either to a central third-party or to other participants. Participants hence need to trust these parties not to either accidentally or voluntarily breach their privacy. Within the scope of this paper, we propose a novel approach that protects participants' location privacy independently of their degree of trust in other parties.

Our contributions can be summarized as follows:

- We present an *OPPortunistic Privacy-Preserving* (OP^4) scheme based on secure multi-party computation [39]. In OP^4, participants located in physical proximity can identify and construct common geographical areas without disclosing their original locations. By reporting such areas instead of their individual locations, the participants protect their location privacy against curious campaign administrators. Nevertheless, this protection is only ensured if the involved participants cooperate and actually report the same area to the application server. We therefore use the concept of secret sharing [33] to guarantee that campaign administrators only have access to the constructed area if a sufficient number of participants have reported it.
- We conduct a thorough threat analysis showing the robustness of OP^4 against privacy threats.
- OP^4 is a distributed scheme that relies on opportunistic physical encounters between participants. We therefore explore its feasibility by undertaking a detailed simulation-based analysis based on a real-world dataset and especially compare its performance against a centralized solution.
- We further evaluate the feasibility of our approach by implementing a proof-of-concept on Android Nexus 6 mobile phones and measure the incurred overheads.

The paper is organized as follows. In Sec. II, we discuss related work. We introduce our assumptions and models in Sec. III and present OP^4 in Sec. IV. We conduct a multi-dimensional evaluation in Sec. V, before making concluding remarks in Sec. VI.

II. Related Work

Crowdsensing applications can threaten the participants' privacy at different levels, ranging from the distribution of sensing tasks to the presentation of campaign results as shown in [3], [9]. While methods to protect the anonymity of the participants have been proposed in, e.g., [8], we focus in this work on threats to location privacy caused by the reporting of spatiotemporal annotations to the application server.

To address this issue, existing approaches consider either individual locations or trajectories [3]. Since OP^4 concentrates on individual locations, we especially compare our solution to similar approaches. In addition to local mechanisms applied

on the participants' device (such as applied in [7]), most approaches are based on the concept of k-anonymity [36] and aim at finding k participants sharing a same region. By sharing and reporting the same region, the k participants become indistinguishable for the campaign administrators. The challenge is thus to find k users and compute the associated region. To this end, different proposals have been made. The first option is to use a central *trusted third-party* (TTP), which is in charge of building user groups and generating the respective region based on the original locations reported by the participants. Different variants of centralized solutions have been introduced in [12], [14], [37], [32]. However, these solutions require participants to disclose their true locations to the TTP. Participants hence need to trust the TTP to respect their privacy, but have no guarantees.

Instead of trusting a third-party, distributed approaches based on collaborating participants have been proposed. For example, users of location-based services can collaborate to protect their privacy by locally caching the server's answers to their queries. As a result, participants only disclose their location when the queries are not answered yet. Alternatively, participants can collaborate to protect their privacy by, e.g., exchanging collected sensor readings in [5], [6] or find $k-1$ other participants and construct a commonly shared region based on either direct ad hoc communication [13], [22] or supported by a peer-to-peer infrastructure [17]. Again, the participants need to disclose their original locations to other participants. When compared to the previously described centralized solutions, the trust issue is not solved, but only distributed to more parties. In contrast, OP^4 is a distributed scheme that leverages cooperation between participants to compute a shared region based on ad hoc communication between devices. The key difference to the above schemes is that participants' locations are compared without being disclosed. Our work is an extension of our previous work [4], but differs in (1) the adopted communication models, (2) the methods applied to build common regions, (3) the conducted simulations, and (4) the datasets used in our simulations. Moreover, no proof-of-concept implementation is proposed in our previous work.

III. Assumptions

We make the following assumptions regarding our system model and the related threats to privacy.

A. System Model

We assume a delay-tolerant crowdsensing application, such as a noise pollution monitoring application. Participants identified by a unique identifier ID collect sensor readings at a given sampling frequency. Each collected sensor reading s is annotated with the time t and the location l of its collection, thus forming a triplet $T = \langle t, l, s \rangle$. We further assume that the participants' mobile phones referred to as clients can establish ad hoc communication when located in physical proximity. The collected triplets are reported to the application server, where they are processed, before being presented to end users. All participants and the application server own a public/privacy key pair.

B. Threat Model

We assume a honest-but-curious adversary model, in which campaign administrators attempt to passively breach the participants' privacy based on their reported triplets. They run the system normally and faithfully, but do not launch active attacks to obtain further information. Due to the distributed nature of OP^4, participants also become possible adversaries, who are potentially interested in inferring other participants' whereabouts. Like for the campaign administrators, we assume that participants are honest-but-curious. Note that campaign administrators and participants do not collaborate to launch colluding attacks.

IV. OP^4 Architecture

We first provide an give a high-level overview of the OP^4 architecture illustrated in Fig. 1, before detailing the underlying mechanisms applied in OP^4 in Sec. IV-B. We assume that the participants follow the system model introduced in Sec. III-A.

A. Overview

The system's typical operation is as follows. Each participant i collects a set of triplets $\{T_j^i = \langle t_j^i, l_j^i, s_j^i \rangle | 1 \leq j \leq n, j \in \mathbb{N}\}$ annotated with spatiotemporal information. n is the total number of triplets collected by i. Let us assume that N participants are in physical proximity (i.e., within communication range) during the collection process. They collaborate to build a common region r including at least k participants and report this region instead of their original location to the application server. By doing so, they ensure their location k-anonymity. When compared to existing schemes, each participant however does not reveal her original locations to the $N-1$ other participants. To this end, the N participants initiate a secure *Multi-Party Computation* (MPC) protocol based on garbled circuits as detailed in Sec. IV-B1. This cryptographic protocol allows the N participants to intersect the set $L^i = \{(t_j^i, l_j^i) | 1 \leq j \leq n, j \in \mathbb{N}\}$ of their visited locations while keeping them secret. A set element (i.e., a visited location) is only disclosed to the participants if it is shared between them.

Given that a set element (t, l) is found to be common to at least k of the N users, a function is applied to generalize the found location to a common region r of radius d shared by the k users. The k users next replace their original location by the computed region r in the triplets to be reported to the application server. To ensure their location k-anonymity, all k users must report their collected sensor readings using this region. Instead, the participants apply a secret sharing algorithm detailed in Sec. IV-B2. In short, each participant encrypts her collected sensor readings and splits the result into $m > k$ shares. She keeps one share for herself and distributes the remaining shares to the $k-1$ users. After a timeout T_{secret}, the k participants report their own share as well as those

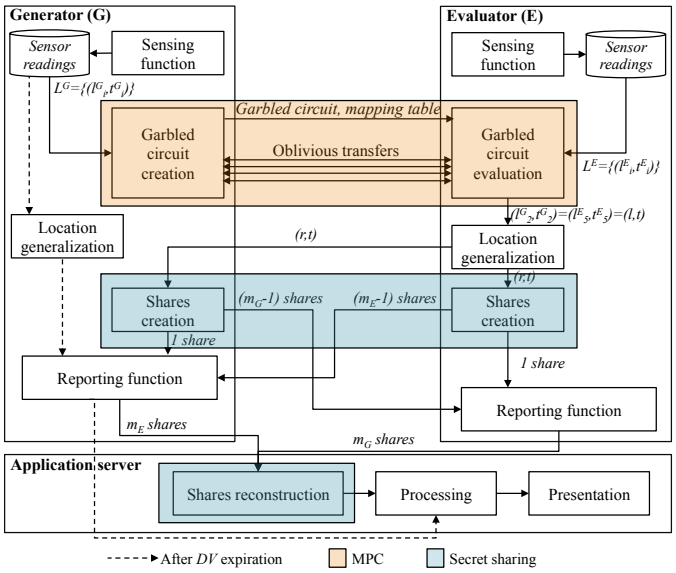


Fig. 1. Overview of the OP^4 architecture given N=2. We assume that the generator reports its sensor readings with generalized locations after T_{report}, while the evaluator discard them

from the remaining participants for each shared location. The application server is only able to decrypt the reported sensor readings when k shares are reported. This property guarantees the participants' location k-anonymity.

The remaining elements, i.e., those that are not shared with others, are retained by the participants to be compared with other participants at future opportunistic encounters. In absence of encounters, the triplets are either suppressed or their location is individually generalized before being reported to the application server after a timeout T_{report}. The decision between suppression and generalization is made by the participants at the beginning of the collection process depending on their individual privacy preferences.

In summary, participants can control their privacy protection by tailoring the following parameters to their preferences: N_{min} the minimum number of participants in physical proximity needed to initiate a MPC protocol, k_{min} the minimum number of participants sharing the same location, and d_{min} the minimum radius of the generalized region around their original locations.

B. Underlying Mechanisms

1) Secure Multi-Party Computation: To compare their locations without disclosing them, the participants leverage secure computation based on garbled circuits, as introduced in [39]. Its goal is to allow two parties to compute the output of a function f based on the inputs of both parties but without each party learning the input from the other. To this end, one of both parties referred to as *circuit generator* first translates f into a Boolean circuit composed of binary gates, each of them being connected by wires. For each gate, the following operations are conducted. To illustrate them, we choose an AND gate as example. The gate has two input wires, x and y, and one output wire z as illustrated in the gate truth table in Tab. I(a).

a) Garbled Circuit Creation: The generator first associates two random secret cryptographic keys (K_w^0, K_w^1) to the respective value of each wire. Tab I(b) shows the resulting key distribution. It then computes the corresponding gate outputs by encrypting them using the aforementioned keys as shown in Tab. I(c). The resulting encrypted outputs are then shuffled and build a so-called garbled gate, which is part of the garbled circuit. Tab. I(d) shows an example of a corresponding garbled circuit, assuming that the garbled circuit contains a unique AND gate, i.e., f is an AND gate.

b) Garbled Circuit Transmission: The constructed garbled circuit is sent to the second party referred to as *circuit evaluator* along with the mapping between the input values and the output-wires keys illustrated in Tab. I(e).

c) Garbled Circuit Evaluation: The evaluator needs the input-wire keys, i.e., (K_x^0, K_x^1) and (K_y^0, K_y^1). To this end, a *one-out-of-two oblivious transfer* (OT_2^1) [10], [30] is initiated for each wire w. In each transfer, the generator inputs the two corresponding keys K_w^0 and K_w^1, while the circuit evaluator inputs a selection bit $b \in \{0, 1\}$. The result v_w^b is then communicated to the evaluator. As a result, the inputs remain secret, i.e., b is kept secret from the generator and $K_w^{\neg b}$ from the evaluator, except if the function output reveals them. For example, the output of an AND gate reveals the inputs when it is equal to 1. The evaluator finally evaluates the circuit by decrypting the garbled truth table (see Tab. I(d)) using the input-wire keys obtained in the oblivious transfers and the mapping transmitted by the generator (see Tab. I(e)).

Consequently, any function f can be translated into a garbled circuit based on binary gates, the output of one gate being one of the inputs of the following gate. To reduce the communication and computation overheads, different optimizations can be applied to the original scheme, such as the point-and-permute technique [25], free-XOR technique [20], or the garbled row reduction [28] scheme. Moreover, the two-parties scheme detailed above can be extended to include additional parties. The design and optimization of secure multi-party computation schemes, however, are beyond the scope of this manuscript.

2) Secret Sharing Algorithm: To ensure the participants' location k-anonymity, we apply the concept of secret sharing [33] using an erasure coding function [29] and articulated into three main steps.

a) Shares Creation: The first step is conducted by each participant having a triplet $T' = \langle t, r, s \rangle$, with r being the generalized location shared with k other users. The triplet is first divided into k shares, which are then transformed into $m > k$ shares by applying an erasure coding function g [31]. Erasure coding allows the application server to reconstruct each triplet as soon as k of the triplet's shares are available at the application server. This caters for missing shares caused by, e.g., transmission errors during the exchange or reporting process or participants opting out. A tag is next appended to each share to later allow the application server to identify

TABLE I
GARBLED CIRCUIT CONSTRUCTION FOR THE BOOLEAN AND GATE

(a) Gate truth table

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

(b) Wire's keys

x	y	z
K_x^0	K_y^0	K_z^0
K_x^0	K_y^1	K_z^0
K_x^1	K_y^0	K_z^0
K_x^1	K_y^1	K_z^1

(c) Encrypted truth table

x	y	z
0	0	$E_{K_x^0}(E_{K_y^0}(K_z^0))$
0	1	$E_{K_x^0}(E_{K_y^1}(K_z^0))$
1	0	$E_{K_x^1}(E_{K_y^0}(K_z^0))$
1	1	$E_{K_x^1}(E_{K_y^1}(K_z^1))$

(d) Garbled truth table

x	y	z
1	1	$E_{K_x^0}(E_{K_y^0}(K_z^0))$
0	0	$E_{K_x^0}(E_{K_y^1}(K_z^0))$
0	1	$E_{K_x^1}(E_{K_y^0}(K_z^0))$
1	0	$E_{K_x^1}(E_{K_y^1}(K_z^1))$

(e) Mapping table

x	y	z
0	0	K_z^0
0	1	K_z^0
1	0	K_z^0
1	1	K_z^1

shares reported by different participants, but belonging to the same triplet. The tag is the result of a cryptographic hash function h applied to a generated random nonce and the corresponding participants' identifier ID. In order to protect the participant's privacy against eavesdroppers and malicious participants, each share concatenated with the generated tag is encrypted using the public key of the application server $k_{pub/server}$. The resulting shares are hence:

$$T_i'' = E_{k_{pub/server}}(g(T')||h(ID, nonce)), \forall 0 \leq i \leq m \quad (1)$$

b) Shares Exchange: Each participant keeps one of the m created shares and exchange the remaining with the k other users involved in the encounter.

c) Shares Reconstruction: We assume that the participants report both their own shares as well as those received from other participants to the application server. The application server decrypts each reported share T_i'' with its private key $k_{pri/server}$:

$$D_{k_{pri/server}}(T_i'') = g(T_i'||h(ID, nonce)) \quad (2)$$

Then, it checks whether additional shares with the same tag, i.e., $h(ID, nonce)$, have already been decrypted. If k shares are available, the original triplet is decoded using the decoding function g^{-1} of the applied erasure coding scheme.

V. EVALUATIONS

We conduct a multidimensional evaluation of OP^4. First, we analyze the protection offered by OP^4 to participants when considering our threat model introduced in Sec. III-B. We then quantify the performance of OP^4 and compare them to those of a centralized approach. We finally measure the introduced overhead using our prototypical implementation.

A. Threat Analysis

In what follows, we discuss the resilience of OP^4 against different attack scenarios and hence offer the following protection to the participants of crowdsensing applications.

1) Protection against Honest-but-Curious Campaign Administrators: They are curious about the participants and their visited locations. However, they do not launch active attacks to gain access to this information. Assuming that a participant has encountered no other OP^4 participants during T_{report} (i.e., the given validity period of the collected sensor readings), she either reports her sensor readings by cloaking her exact locations or discards them depending on her personal privacy preferences. Additionally, the secret sharing algorithm guarantees that k users have reported their sensor readings collected in the same region, thus ensuring their location k-anonymity. It also breaks the link between the collected sensor readings and the identity of the participants having collected them. As a result, the campaign administrators do not gain access to the original and exact locations of the participants. The degree of privacy protection depends on the number of k encountered participants and the applied cloaking scheme, i.e., the size of the generalization area.

2) Protection against Honest-but-Curious Participants: Participants collaborate in OP^4 to protect their privacy. Due to the MPC application, the participants do not reveal their locations to each other in the comparison process. Only if participants have locations in common, the MPC algorithm will disclose them to the respective participants in order to compute the common cloaked region. In this case, the protection depends on the applied MPC approach, especially on the oblivious transfer protocol implementation. Well-established oblivious transfer protocols are usually secure against honest-but-curious parties ([25], [15]). For example, a trivial implementation of an oblivious transfer is proposed in [25], in which two parties A and B are involved. A first generates two messages for B: (1) a public key k_{pub} of a public/private key pair she knows and (2) a random string s. B then encrypts both messages using the original messages as key. The results $E_{k_{pub}}(k_{pub})$ and $E_s(s)$ are sent back to A. As a result, A is only able to decrypt one of both results. Note that more complex implementations can be adopted to ensure protection against malicious participants launching active attacks is also possible in oblivious transfers as proposed in [23], [26], [24], [34]. This additional protection, however, leads to increased overheads.

In summary, OP^4 protect the participants' location privacy against both honest-but-curious participants and campaign administrators.

B. Performance

We next measure the performance of OP^4 based on extensive simulations.

1) Settings: We use the *Mobile Data Challenge* (MDC) dataset as underlying participants' mobility model [19], [21]. It contains the mobile phone data of 185 participants collected between September 2009 and March 2011 in Switzerland.

Algorithm 1 Spatial preprocessing algorithm

Require: l_i: location at time t_i,
l_{i+1}: location at time t_{i+1},
l_{i+2}: location at time t_{i+2},
$\delta t_{i+1-i} = t_{i+1} - t_i$: time difference,
ΔT: decision threshold,
n_T: total number of triplets
for each iteration i with $i \in [1, n_T]$ **do**
  if $i = 1$ **or** $\delta t_{i+1-i} \geq \Delta T$ **then**
    l_i=l_i ▷ Start point S: unmodified
  else if $i + 1 = n_T$ **or** $\delta t_{i+2-i+1} \geq \Delta T$ **then**
    l_i=l_i ▷ End point E: unmodified
  else if $\delta t_{i+1-i} = 0$ **then**
    $l_i = \frac{l_{i-1}+l_{i+1}}{2}$ ▷ Duplicate D: replaced
  else
    $l_i = \frac{\delta t_{i-i-1} \cdot l_{i-1} + \delta t_{i+1-i} \cdot l_{i+1}}{\delta t_{i-i-1} + \delta t_{i+1-i}}$ ▷ Midpoint M: replaced
  end if
end for

In total, 11,035,020 GPS samples are available during this period (after having removed duplicates). The number of daily collected GPS samples ranges between 812 and 49,294 (mean μ=19) involving three to 101 participants (μ=59).

We normalize the GPS samples using an interval of 5s, as the sampling rate depends on the device's battery level and hence, is not uniformly distributed. To this end, we preprocess the GPS data by applying a binning technique, i.e., replacing each value within a bin by the time corresponding to the bin's lower bound as illustrated in Fig. 2(a) and Fig. 2(b). To reduce the spatial errors introduced by the temporal preprocessing, we apply Alg. 1. The algorithm first distinguishes the category of each GPS sample between: (1) a starting point S of a new trajectory, (2) a duplicate D (i.e., different participant's locations in the same bin), (3) a midpoint M within a trajectory, or (4) an end point E terminating a trajectory. Depending on the identified category, the original sample is either kept or modified. The labels in Fig. 2(b) illustrate the recognized categories for different samples, while Fig. 2(c) shows the resulting samples after the application of the spatiotemporal transformation. Finally, Fig. 2(d) visualizes the differences between the original and transformed trajectories.

To identify encounters of at least k users, we further apply Alg. 2 to the preprocessed dataset. The algorithm first relies on the construction of a multi-dimensional space partitioning Ball Tree structure [27], [18]. The constructed Ball Tree is then queried for each sample's neighbor with the communication range as query radius. The list of neighbors is reduced to optimize the resulting cluster size and avoid that samples belong to more than one cluster. As a result, the algorithm generates a set of n clusters containing at least k users.

2) Results: We investigate and comment on the performance of OP[4] depending on different configuration parameters.

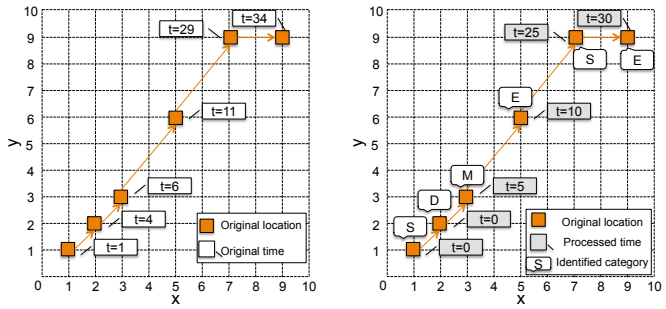
a) Number of Encounters: Participants can set the minimal number N_{min} of other participants involved in an encounter to start the MPC protocol. The larger the number, the higher the probability to find participants sharing the same location and hence, the better for the privacy protection. Simultaneously, the larger the number, the lower the probability that such encounter happens. Additionally, the MPC protocol consumes resources as shown in Sec. V-C. Hence, a *Cool-down Time* CT between two consecutive executions (i.e., encounters) is necessary to save battery lifetime. We therefore explore based on the above simulation settings the impact of both N and CT on the number of encounters. To this end, we assume that the ad hoc communication between the participants is based on either Bluetooth or Wi-Fi and is possible within a range of 100 meters [1].

Algorithm 2 Spatial Clustering

Require: L: set of locations l_i,
r_{com}: communication range,
k: minimum number of users in a group
tree ← BallTree (L); ▷ Construct a BallTree object
neighbors ← tree.query_radius(L, d_{com}); ▷ Get the nearest neighbors within r_{com} for each l_i
neighbors_sets ← list(set(s), $\forall s$ in neighbors);
intermediate_clusters ← list();
for $i \leftarrow 0$ **to** length(neighbors_sets) **do** ▷ Find the minimal neighbors' subset
  $l \leftarrow$ neighbor_sets[j], $\forall j \in$ neighbor_sets[i]; ▷ Get the list of current neighbors' neighbors
  group ← $\bigcap_{x \in l} x$; ▷ List reduction by set intersection
  if length(group) $\geq k$ **then**
    intermediate_clusters.append(group);
  end if
end for
assigned_nodes ← set();
final_clusters ← list();
while length(intermediate_clusters) > 0 **do**
  max ← a cluster of maximal size;
  if max $< k$ **or** length(assigned_nodes) = length(L) **then**
    break;
  end if
  reduced_group ← max $\setminus$ assigned_nodes; ▷ Inclusion of locations not in another cluster yet
end while
if length(reduced_group) $> k$ **then**
  final_clusters.append(reduced_group);
  assigned_nodes ← assigned_nodes $\cup$ reduced_group;
end if
intermediate_clusters ← intermediate_clusters $\setminus$ max;
groups← $\bigcup_{s \in final\_clusters}$;

Fig. 3 demonstrates the feasibility of our approach, as participants are involved in more than 100,000 meetings for $N = 2$ and $CT = 5s$. As expected, we observe that increasing N significantly reduces the number of meetings. Note that no meetings happen for $N \geq 6$. The value of CT especially impacts the number of encounters for $CT \leq 500s$. For greater CT values and the same N value, the number of meetings remains almost constant. This can be explained that the greater CT, the lower the probability that participants located in the same area will stay and be involved in further meetings.

b) Number of k-anonymous Locations: The participants can collaborate to protect their locations if (1) they are in physical proximity and (2) share a common visited location. This physical proximity requirement adds an additional con-



(a) Original participant's trajectory (b) Temporal transformation

(c) Spatiotemporal transformation (d) Comparison of both trajectories

Fig. 2. Example of the spatiotemporal transformation resulting from the application of Alg. 1 on a given trajectory ($\Delta T = 10$)

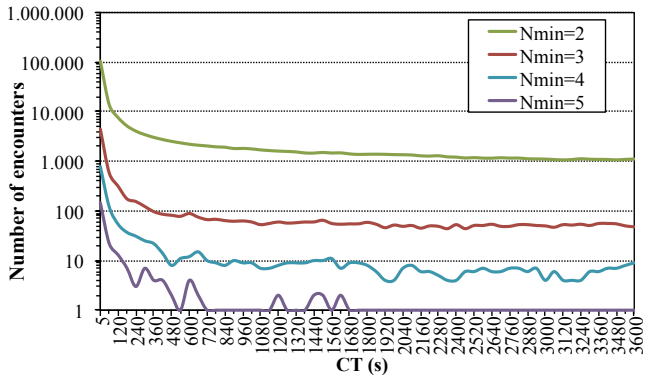


Fig. 3. Impact of the minimum number of participants N_{min} and the cool-down time CT on the number of identified encounters

straint as compared to a centralized solution, which is an optimal solution as the third-party knows all exact participants' locations and can hence construct the greater number of k-anonymous locations due to its global view. In contrast, the participants have only a local view restricted to the number of encountered participants in OP^4. We therefore expect OP^4 to identify a lower number of k-anonymous locations as compared to a centralized approach. To quantify the difference between OP^4 and a centralized approach, we run Alg. 2 to build clusters of at least k users within a given range to simulate a centralized solution.

We present the ratio of the number of k-anonymous locations identified using OP^4 over the same number identified with the centralized approach in Fig. 4. For our simulations, we choose the cool-down time $CT = 5s$, the data validity time $T_{report} = 24$ hours, i.e., the time after which the sensor readings are either individually reported to the server after local cloaking or discarded depending on the participants' preferences. We vary the required number of participants per meeting N, the minimum participants' number k necessary to release a common location, and the range within which the participants should be to share the same location in the centralized approach.

For $N \leq k$ and a range of 100 m, our approach shows almost the same number of k-anonymous locations than a centralized solution. For all k and N values, the performance degrades when the given range augments. In these cases, the centralized solution outperforms ours, as additional clusters can be built between users that did not meet according to our conditions. A similar observation can be made when $k < N$. The centralized approach is able to identify more participants within the same range as compared to OP^4, which first requires a meeting between at least N participants. Again, the greater N, the more difficult to find potential meeting candidates. Similarly, the greater k, the more difficult to find participants to build a group in the same region. As a result, the performance of the centralized approach diminishes with an increase of both N and k, converging to the performance of OP^4.

c) Reporting Overhead: By design, the participants sharing a common location divide the triplets to be reported into shares according to the secret sharing algorithm detailed in Sec. IV-B2. The division is based on the number of participants involved in the process and the number of shares $m > k$ generated by the participants. Each participant keeps one of her own shares and distributes the remaining to the others. We have therefore measured the introduced overhead for the participants, i.e., the number of exchanged shares reported to the application server as compared to the initial number of collected sensor readings. In almost all cases, the overhead is zero for the participants meaning that the shares are equally distributed between all participants. In very few cases, one participant needs to report more sensor readings than in absence of OP^4. This happens, e.g., when a participant has only one location common to the others, while the $k-1$ other participants have many locations in common, thus generating many more shares than her own. Thus, the shares' reporting process remains fair for a majority of participants.

In summary, we have shown based on a real-world dataset that participants are able to compare their locations based on ad hoc communication and find common locations to build a shared region with a reporting overhead evenly distributed between them. As a result, we have confirmed the feasibility of our approach.

C. Proof-of-Concept

We have built a prototypical implementation of OP^4 to quantify the incurred resource consumption for the participating devices. Our proof-of-concept is based on the two-party computation framework tailored to mobile devices proposed in [15]. We hence leverage an existing garbled circuit for

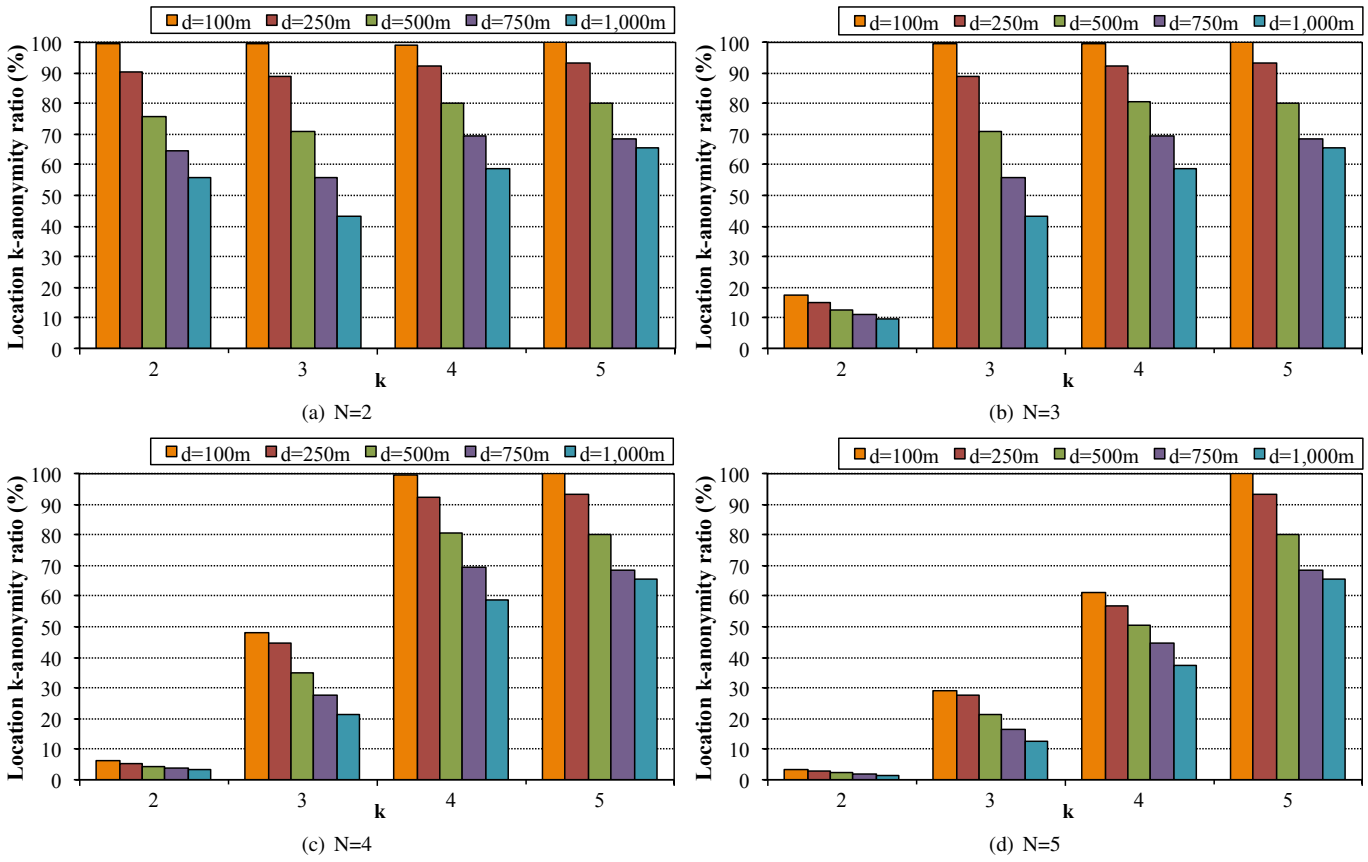


Fig. 4. Impact of the number N of participants involved in meetings, the number of k participants sharing the same location, and the radius d on the ratio of k-anonymous locations released by OP^4 as compared to a centralized approach

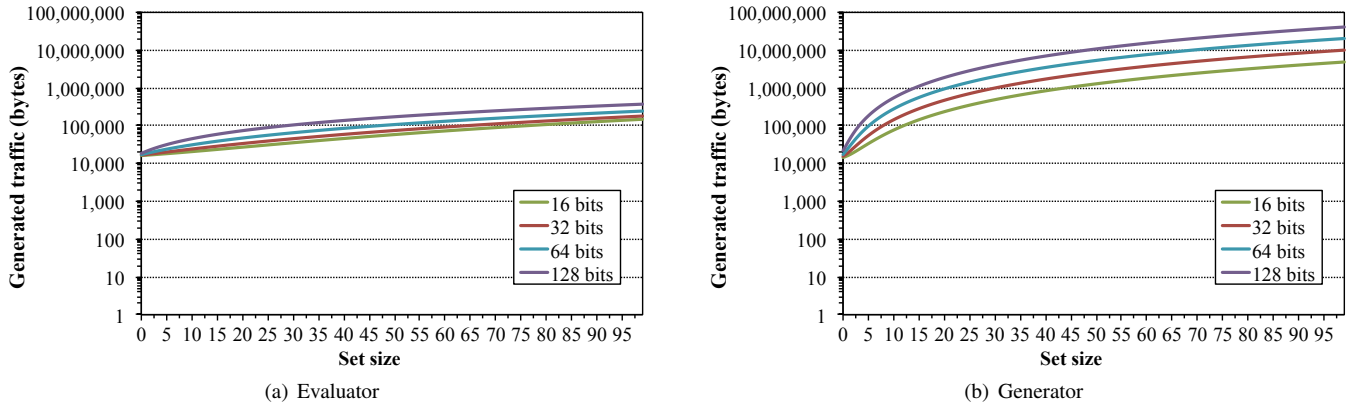


Fig. 5. Impact of the set size Φ and the element size λ on the amount of generated traffic by the evaluator and generator, respectively

set intersection and have adapted it to run on two Nexus 6 phones featuring Android operating system version 5.1.1. Each device has quad-core CPUs with a clock speed of 2.7 GHz, 3 GB of RAM and 64 GB of internal memory. To balance the computing load between the devices, each device alternatively plays the role of generator and evaluator in the MPC algorithm (see Sec. IV-B1). Using our prototype, both devices can discover each other and compare two sets of different sizes to find common elements without disclosing them. The length of each set element depends on the application scenario. In our case, we assume that the timeout for reporting data T_{report} is 24 hours. This means that all data are reported to the application server exclusively within this period. Moreover, we adopt a sampling rate of 5s common to all contributing users, allowing us to encode the timestamp over 11 bits. The location, i.e., latitude and longitude, can be encoded using 26 bits when reducing the accuracy to 1 m. As a result, each set element, i.e., time and location, can be encoded using 64 bits. Alternatively, both participants can first disclose and compare their timestamps, before starting

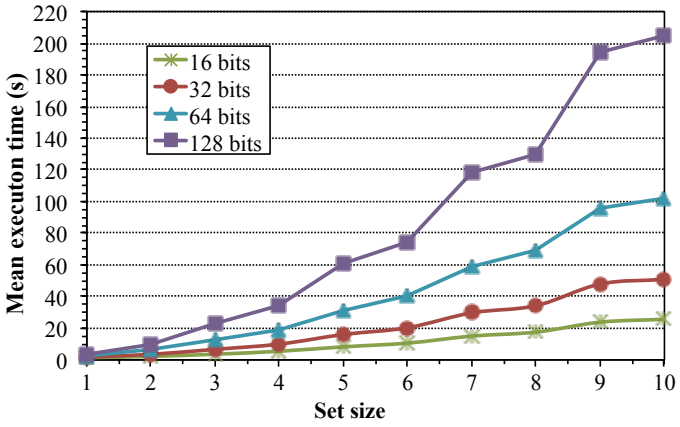


Fig. 6. Impact of the set size Φ and the element size λ on the execution time

the MPC algorithm to compare their locations in case of common timestamps. This would reduce the set intersection to 32-bit elements. To also cater for additional scenarios, we hence present the performance of our proof-of-concept implementation for set elements of lengths: 16, 32, 64, and 128 bits. Note that additional methods to optimize the selection of the set elements to be compared in the MPC protocol could be proposed, e.g., based on the population density at the collection locations in order to further reduce the incurred overheads. Their design is however considered as future work.

1) Communication Overhead: We first measure the traffic generated by the generator (see Fig. 5(b)) and the evaluator (see Fig. 5(a)), respectively. To this end, we vary the set size Φ (i.e., the number of compared elements) as well as the element length λ. As a result, we observe that comparing two sets of 50 64-bit elements generates 10.54 MB of traffic in total. This includes 0.17 MB generated by the evaluator and 10.37 MB generated by the generator. The traffic difference between both entities is caused by the task asymmetry between both roles. Recall from Sec. IV-B1 that the generator is responsible for generating the garbled circuit and then for transmitting it to the evaluator. In contrast, the evaluator only evaluates the transmitted circuit and sends the results back to the generator. In this case, the data transmission hence takes around 3s for participants communicating via Bluetooth version 4.0 and 0.3 s for participants leveraging a Wi-Fi direct communication [38].

2) Execution Time: We finally measure the execution time for different values of Φ and λ. The execution time includes (1) the computation of the set intersection, (2) the transmission of the garbled circuits to the evaluator, and (3) the transmission of the results from the evaluator to the generator. Fig. 6 shows the mean computation time for one intersection computed over 50 runs (SD=3.5s). The runs are equally distributed between battery-powered and AC-powered devices. Based on our above assumptions, the execution of a comparison of two sets of ten 64-bit elements takes approximately 100s. Nevertheless, the obtained values highly depend on the memory management in the Android operating system. To further reduce the execution time, the MPC framework detailed in [15] should be optimized by explicitly managing the memory in order to avoid interruptions introduced by the garbage collector. Such improvements are considered as future work.

VI. Conclusions

We have presented a distributed approach based on opportunistic encounters between participants of crowdsensing applications. Our approach called OP^4 protects the location privacy of participants reporting sensor readings to the application server. As compared to existing work, the participants do not need to trust any other parties in OP^4 to protect their privacy, as their locations are only revealed to the participants also sharing them. These participants then collaborate to report their sensor readings to the application server and prevent the campaign administrators from inferring their originally visited locations. We have evaluated OP^4 based on a real-world dataset serving as underlying mobility model for potential participants. We have first shown its feasibility and compared our approach to a centralized solution. Additionally, we have built a proof-of-concept implementation to validate our approach. In the future, different optimizations could be applied to further reduce the resource consumption of the proposed scheme. This includes an improved explicit management of the memory in the MPC framework proposed in [15] and used in our prototype implementation. Moreover, more efficient MPC algorithms, such as [40], [16], could be integrated in the framework. Finally, an optimized scheme to encode the spatiotemporal information could be applied in order to reduce the length of the set elements to be compared.

VII. Acknowledgements

Portions of the research in this paper used the MDC Database made available by Idiap Research Institute, Switzerland and owned by Nokia.

References

[1] "Bluetooth Core Specification 4.2," Online: https://www.bluetooth.org/en-us/specification/adopted-specifications (accessed in 09.15), 2014.

[2] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory Sensing," in *Proc. of the 1st Workshop on World-Sensor-Web (WSW)*, 2006.

[3] D. Christin, "Privacy in Mobile Participatory Sensing: Current Trends and Future Challenges," *Journal of Systems and Software (JSS)*, 2015.

[4] D. Christin, D. Bub, A. Moerov, and S. Kasem-Madani, "A Distributed Privacy-Preserving Mechanism for Mobile Urban Sensing Applications," in *Proc. of the 10th IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2015.

[5] D. Christin, J. Guillemet, A. Reinhardt, M. Hollick, and S. S. Kanhere, "Privacy-preserving Collaborative Path Hiding for Participatory Sensing Applications," in *Proc. of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2011.

[6] D. Christin, D. R. Pons-Sorolla, M. Hollick, and S. S. Kanhere, "TrustMeter: A Trust Assessment Framework for Collaborative Path Hiding in Participatory Sensing Applications," in *Proc. of the 9th IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014.

[7] D. Christin, C. Roßkopf, and M. Hollick, "uSafe: A Privacy-aware and Participative Mobile Application for Citizen Safety in Urban Environments," *Pervasive and Mobile Computing (PMC)*, vol. 9, no. 5, 2013.

[8] D. Christin, C. Roßkopf, M. Hollick, L. A. Martucci, and S. S. Kanhere, "IncogniSense: An Anonymity-preserving Reputation Framework for Participatory Sensing Applications," in *Proc. of the 10th IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2012.

[9] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick, "A Survey on Privacy in Mobile Participatory Sensing Applications," *Journal of Systems and Software*, vol. 84, no. 11, 2011.

[10] S. Even, O. Goldreich, and A. Lempel, "A Randomized Protocol for Signing Contracts," *Communications of the ACM*, vol. 28, no. 6, 1985.

[11] R. K. Ganti, F. Ye, and H. Lei, "Mobile Crowdsensing: Current State and Future Challenges," *IEEE Communications Magazine*, vol. 49, no. 11, 2011.

[12] M. Gruteser and D. Grunwald, "Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking," in *Proc. of the 1st International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2003.

[13] T. Hashem and L. Kulik, "Safeguarding Location Privacy in Wireless Ad-Hoc Networks," in *Proc. of the 9th International Conference on Ubiquitous Computing (UbiComp)*, ser. Lecture Notes in Computer Science, J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, Eds. Springer Berlin Heidelberg, 2007, vol. 4717.

[14] K. L. Huang, S. S. Kanhere, and W. Hu, "Preserving Privacy in Participatory Sensing Systems," *Computer Communications*, vol. 33, no. 11, 2010.

[15] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster Secure Two-party Computation Using Garbled Circuits," in *Proc. of the 20th USENIX Conference on Security (SEC)*, 2011.

[16] Y. Huang, J. Katz, and D. Evans, "Efficient Secure Two-Party Computation Using Symmetric Cut-and-Choose," in *Advances in Cryptology - CRYPTO*, ser. Lecture Notes in Computer Science, R. Canetti and J. Garay, Eds. Springer Berlin Heidelberg, 2013, vol. 8043.

[17] L. Kazemi and C. Shahabi, "A Privacy-aware Framework for Participatory Sensing," *SIGKDD Explor. Newsl.*, vol. 13, no. 1, 2011.

[18] A. M. Kibriya and E. Frank, "An Empirical Comparison of Exact Nearest Neighbour Algorithms," in *Proc. of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, 2007.

[19] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila, "Towards Rich Mobile Phone Datasets: Lausanne Data Collection Campaign," in *Proc. of the ACM International Conference on Pervasive Services (ICPS)*, 2010.

[20] V. Kolesnikov and T. Schneider, "Improved Garbled Circuit: Free XOR Gates and Applications," in *Proc. of the 35th International Colloquium on Automata, Languages and Programming (ICALP)*, 2008.

[21] J. K. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen, "The Mobile Data Challenge: Big Data for Mobile Computing Research," in *Proc. of the Mobile Data Challenge Workshop (MDC) in conjunction with the International Conference of Pervasive Computing (Pervasive)*, 2012.

[22] Q. Li and G. Cao, "Efficient and Privacy-preserving Data Aggregation in Mobile Sensing," in *Proc. of the 20th IEEE International Conference on Network Protocols (ICNP)*, Oct 2012.

[23] Y. Lindell and B. Pinkas, "An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries," in *Advances in Cryptology - EUROCRYPT 2007*, ser. Lecture Notes in Computer Science, M. Naor, Ed. Springer Berlin Heidelberg, 2007, vol. 4515.

[24] ——, "Secure Two-Party Computation via Cut-and-Choose Oblivious Transfer," in *Theory of Cryptography*, ser. Lecture Notes in Computer Science, Y. Ishai, Ed. Springer Berlin Heidelberg, 2011, vol. 6597.

[25] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay-Secure Two-Party Computation System." in *Proc. of the 13th Conference on USENIX Security Symposium (SSYM)*, 2004.

[26] J. Nielsen and C. Orlandi, "LEGO for Two-Party Secure Computation," in *Theory of Cryptography*, ser. Lecture Notes in Computer Science, O. Reingold, Ed. Springer Berlin Heidelberg, 2009, vol. 5444.

[27] S. M. Omohundro, "Five Balltree Construction Algorithms," International Computer Science Institute, Berkeley, California, Tech. Rep., 1989.

[28] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, "Secure Two-Party Computation Is Practical," in *Advances in Cryptology - ASIACRYPT*, ser. Lecture Notes in Computer Science, M. Matsui, Ed. Springer Berlin Heidelberg, 2009, vol. 5912.

[29] F. Qiu, F. Wu, and G. Chen, "SLICER: A Slicing-Based K-Anonymous Privacy Preserving Scheme for Participatory Sensing," in *Proc. of the 10th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, Oct 2013.

[30] M. O. Rabin, "How to Exchange Secrets with Oblivious Transfer," Aiken Computation Lab, Harward University, Tech. Rep. TR-81, 1981.

[31] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, 1960.

[32] I. Rodhe, C. Rohner, and E. C.-H. Ngai, "On Location Privacy and Quality of Information in Participatory Sensing," in *Proc. of the 8th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet)*, 2012.

[33] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, no. 11, 1979.

[34] A. Shelat and C.-H. Shen, "Two-Output Secure Computation with Malicious Adversaries," in *Advances in Cryptology - EUROCRYPT*, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed. Springer Berlin Heidelberg, 2011, vol. 6632.

[35] K. Shilton, "Four Billion Little Brothers?: Privacy, Mobile Phones, and Ubiquitous Data Collection," *Communications of the ACM*, vol. 52, no. 11, 2009.

[36] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, 2002.

[37] K. Vu, R. Zheng, and J. Gao, "Efficient Algorithms for K-anonymous Location Privacy in Participatory Sensing," in *Proc. of the 31th IEEE Conference on Computer Communications (INFOCOM)*, 2012.

[38] Wi-Fi Alliance, "Wi-Fi Direct," Online: https://www.wi-fi.org/discover-wi-fi/wi-fi-direct (accessed in 09.15).

[39] A. C. Yao, "Protocols for Secure Computations," in *Proc. of the 23rd Annual Symposium on Foundations of Computer Science (SFCS)*, vol. 82, 1982.

[40] S. Zahur, M. Rosulek, and D. Evans, "Two Halves Make a Whole - Reducing Data Transfer in Garbled Circuits Using Half Gates," in *Advances in Cryptology - EUROCRYPT*, 2015.