Computer Networks 120 (2017) 28–42

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Private mobility-cast for opportunistic networks

Gianpiero Costantino[a,*], Rajib Maiti[a], Fabio Martinelli[a], Paolo Santi[a,b]

[a] IIT-CNR, Pisa, Italy
[b] Senseable City Lab, Massachusetts Institute of Technology, Cambridge, MA, US

ARTICLE INFO

Article history:
Received 4 May 2016
Revised 11 October 2016
Accepted 7 April 2017
Available online 8 April 2017

Keywords:
Opportunistic routing
Mobility-cast
Human mobility
Privacy
Secure-Two party computation

ABSTRACT

In this paper, we introduce the notion of mobility-cast in opportunistic networks, according to which a message sent by a user *S* is delivered to users with a mobility pattern similar to that of *S* – collectively named *place-friends*. Motivation for delivering a message to place-friends stems from the fact that current social acquaintances are likely to be place-friends. Most importantly, it has been recently found that a large fraction of *new* social contacts come from place-friends. After introducing mobility-cast, we present a privacy-preserving mobile-cast protocol based on secure two-party computation. The effectiveness of the protocol in delivering messages to place-friends is demonstrated by means of analysis and extensive simulations based on a realistic mobility model. In the last part of the paper, we present two alternative implementations of mobility-cast on the Android platform, and test their computational performance on a number of different smartphones. Overall, the results presented in this paper show that privacy-preserving mobility-cast can be effectively implemented with current mobile phone technology.

1. Introduction

Over the last years, people has increasingly used Social Networks and APPs to strengthen existing, as well as creating new, social ties. For instance, Tinder is a very successful APP that allows people to date nearby users. Tinder is increasingly growing its user base, which has recently hit 10 millions.[1] Like Tinder, other similar emerging phenomena are the clear signal that opportunistic communications are not anymore only a research fact, but that they are likely to increasingly become a business opportunity for information and communication companies. However, a weak aspect of most mobile social applications is the superficial interest of developers in user privacy. An attack[2] on Tinder showed how it is possible to discover the exact latitude and longitude position of people by creating three fake users and making a triangulation of each of them using only the distance to the victim. Thus, Tinder released a new version to patch and the attack that cannot be produced anymore.

The above example highlights a major challenge that developers are facing within the mobile social networking application field: on one hand, use of personal information such as position, interests, gender, etc., is useful to improve the effectiveness of the social networking application in delivering information only to relevant users; on the other hand, directly exposing and exchanging such sensible information between users might cause very substantial privacy violations.

In this paper, we address the above described issues by introducing an innovative information dissemination mechanism, called *mobility-cast*, that embeds a privacy-preserving implementation for our mobility-cast protocol. The novel routing mechanism, which is at the core of mobility-cast, focuses on delivering a message *M* generated by a source user *S* to users who display a mobility pattern similar to that of *S*. Following [1], in this paper we call such set of users the *place-friends* of user *S*. As described in greater detail in the following, mobility-cast finds its motivation in the observation that not only current social acquaintances are likely to be place-friends, but also a large fraction of new social contacts comes from place-friends. The mobility-cast protocol that we introduce, which we call MC2H since information is propagated only up to the second communication hop, is built upon a private function to estimate place-friendships between two users. As carefully analyzed in the paper, the function is private in the sense that, after its execution, a party only acquires minimal information about the other party's mobility profile. The amount of information disclosed to the other party can be controlled through a design parameter of the protocol.

This paper comes as extension of our previous work [2], and the most important differences are:

* Corresponding author.
E-mail addresses: name.surname@iit.cnr.it, gianpiero.costantino@iit.cnr.it (G. Costantino).

[1] http://www.wired.com/2014/04/tinder-valuation/

[2] http://threatpost.com/tinder-patches-vulnerability-that-exposed-user-locations/104398 .

http://dx.doi.org/10.1016/j.comnet.2017.04.010
1389-1286/

- The validation of the Mobility-cast protocol using a new extensive set of simulations based on a realistic mobility model, including investigating the scalability with network size. The new simulations are based on synthetic traces. Them allows flexibility in setting network parameters and a very thorough analysis of mobility-cast performance in a wide set of configurations, including scenarios with increasing network size. The results of new extensive simulation experiments confirm that MC2H strikes the best compromise between coverage, precision, and cost, amongst the protocols evaluated in the experiments.
- The development of a new implementation of the mobility-cast protocol for Android platform, based on the recently introduced Secure-Two party Computation framework CBMC-GC [3]. The new implementation has been developed from scratch porting CBMC-GC into smartphones, and adding to it our forwarding primitive. Computational results have been collected and compared with the application that we have previously developed and reported in the conference version of the paper [2], showing that the new prototype gets an improvement in the order of 70% in terms of computational time.
- The "Introduction" and "Motivation" sections have been deeply reworked to provide new scenarios that motivates our work, highlighting new challenges on mobile social-networking applications. Also, the "Related Work" Section has been suitably extended to mention newer works related to our study.

The paper is structured as follows. Section 2 explains the motivation of our work. In Section 3, we discuss relevant papers in related areas. In Section 4, we define *Place-Friend* as social-forwarding condition. In Section 5, we propose the Mobility-Cast protocol, and in Section 6 we analyse the Mobility-Cast protocol from privacy-leakage point of view. In Sections 7, we discuss the human-mobility model, the metrics that we use to simulate our Mobility-Cast, the simulation parameters, and we show and discuss the findings on Mobility-Cast. In Section 8, we present a prototype implementation of Mobility-Cast for Android Smartphones that uses Secure-Two-party Computation. Finally, Section 9 concludes the paper.

2. Motivation

We introduce a routing protocol for opportunistic networks that delivers a copy of message M generated by user A to all nodes in the network with a mobility pattern similar to A. In accordance with [1], we call the set of nodes with a mobility pattern similar to node A the *place-friends* of node A. How to formally define a user's mobility pattern and a similarity metric between mobility patterns (and, hence, the set of place-friends) is deferred to later sections. However, we anticipate here that "similar mobility patterns" does not mean that place-friends have same trajectories, but only that they share places or zones of a geographical area. We call *mobility-cast* the communication primitive that delivers a copy of the message to place-friends.

Mobility-cast is motivated by the observation that social interactions often occur between individuals with similar mobility patterns: e.g., colleagues who work in the same place, friends attending the same fitness class, etc. This intuitive observation is quantitatively evaluated in [4], where the authors show that social ties between people can be inferred with a good accuracy from co-occurrence in time and space. Hence, delivering a message to node A's place-friends is likely to reach many relevant social ties of node A. More importantly, it has been recently shown [1] that a large fraction (about 30%) of *new* social interactions arise between place-friends. Similar observations have been done in [5], where it is shown that individuals with similar mobility patterns are likely to be close in the social network graph formed of the phone calls between users. Thus, delivering a message to place-friends is useful not only to reach current social ties, but also *forthcoming* social ties. Indeed, we can imagine that a mobility-cast primitive might even increase the fraction of social interactions between place-friends well beyond the 30% value observed in [1]. Suppose individual B, who is a stranger but place-friend of node A, receives an interesting message M from A (e.g., announcing a special event in which B is very interested); having received M, node B might be stimulated to initiate a direct, social interaction with node A.[3]

Since mobility-cast can be used to deliver messages to current, as well as future, social ties, its impact covers the context of mobile social networking and its applications are numerous. For instance, mobility-cast can be used by a fully distributed, opportunistic mobile social networking application to effectively disseminate a message to a selected partition of users, who share same location places, without flooding the network. Or, mobility-cast can be used to implement fully-distributed, opportunistic "friend recommendation" services for social networking applications. Moreover, our routing protocol could be useful to quickly send messages to people during emergency scenarios that involve specific geographical areas.

Overall, in our contribution, all the aforementioned applications are thought to work with respect to users' privacy, In fact, our mobility-cast dissemination protocol contains a privacy-preserving solution that works at the application level and is employed to establish if two users, who are in contact, are *place-friends* without disclosing out personal information.

3. Related work

The choice to use users' social profiles to drive the information propagation process in opportunistic networks has been already investigated in the literature. In [6,7], the authors show that the effectiveness of forwarding an unicast message to destination is improved by considering user social metrics, such as centrality in the social network graph. Mei at al. in [8] propose to exchange users' interest profiles to deliver messages only to sets of interested users. A work which is closer in spirit to ours is [9], where the authors propose to use the user mobility profile to drive message forwarding. However, here a unicast message is sent to a specific destination, while our primitive aims at implementing a communication scheme in which destinations are not known a priori, but are determined by similarity in mobility patterns. Furthermore, privacy issues are not taken into account in [9], and mobility profiles are exchanged among unknown users. Privacy is not considered also in the above mentioned protocols [6–8]. A similar approach for message exchange has been proposed by Hsu et.al. in [10], where mobility of each user is summarized in a $t \times n$ matrix (t and n are the number of time slots and the number of distinct locations respectively) to capture time-aware visit preferences of the user that can visit to a set of n globally agreed locations. Because of huge size of such a matrix, users exchange the eigen vectors of the respective matrices along with a weight value (obtained by singular value decomposition of the matrix) that quantifies an importance level of the locations indicated in the vector. Thus, a message exchange between a pair of users takes place only after a vector similarity computation, which is done by using *weighted sum of inner products* [11] of the respective vectors, and hence privacy of user mobility has not been considered in the proposed scheme.

Privacy-preserving protocols for opportunistic networks are closely related to our work. Most of existing approaches focus

[3] Notice, though, that this would require node A to include his/her identity in M, which might be at odds with the need of preserving privacy.

on securely computing whether two mobile users are "friends", and the specific definition of friendship depends on the approach at hand [12–14]. Only a few protocols consider network-wide information propagation protocols built on top privacy-preserving "friendship" estimation. An example is [15], where the authors introduce a privacy-preserving protocol for geo-casting a message to a specific geographic location. In [16], the authors of this paper present a privacy-preserving version of the interest-casting primitive introduced in [8], including an analysis of the *Privacy-preservation* vs. *Forwarding accuracy* tradeoff which is inherent in the design. An implementation of privacy-preserving interest-casting based on the MobileFairPlay framework has been also presented [17].

In [18], Mohmoud et al. propose a location privacy-preserving solution within the Hybrid Ad Hoc Wireless network model. Here, devices use mobile and fixed network infrastructure to deliver packets from source to destination nodes. The privacy solution proposed in [18] allows communicators to preserve their identities, while secure communications are achieved by means of symmetric-key-cryptography that are stored inside an external Trusted Party. Privacy-preservation while communicating with other users has been proposed in many other related networking scenarios, like mobile social networks [19–21] in order to achieve profile security. While communication among the users in these networks is achieved in a peer-to-peer fashion, a trusted third party is employed in order to ensure unique and secure profile creation, and authentication and identification for the users inside the network. For example, a trusted central authority is assumed in [20] that creates profiles, pseudonyms, and associated certificates for the users in the network. A communication between any pair of users can take place only after proper verification of the certificates done at the central authority.

Similarly, the authors in [21] has utilized Paillier cryptosystem [22] where each user is assumed to be assigned a unique Paillier public/private key pair. A number of encryption and decryption is used to achieve different levels of security to protect user profile for different purposes, e.g., neither sender nor receiver should know attribute values, except the result of the comparison, or neither of them should know the attributes in the intersection set. The authors in [19] have proposed two schemes for private set-intersection (PSI) and private cardinality of set-intersection (PCSI) that heavily rely on secrete polynomial evaluation; each user profile is represented in a polynomial which is then considered for computing intersection set, or intersection set-size. However, a different network scenario is considered where every user first finds the best forwarding user by comparing its own profile with the profiles of every other users in the network before sending the message to it. We use a similar technique, secure two party computation, for privacy preservation of user profile in our work where we are interested only in the intersection set-size (considered a relatively higher level of security), without looking for the intersection set itself.

In [23], Zhang et al. propose a solution similar to ours to preserve *routing metric* information. They contextualize the privacy-preserving solution within the vehicular networks, and they define routing parameters, such us distance or visit frequency to the destination, as *routing metrics*. The goal of the authors is to preserve these metrics from unknown vehicles during the routing phase. This is achieved by using an existing solution for secure two-party computation based on homomorphic encryption. The authors of [24] take into account geocasting issues in vehicular networks. Here, vehicles send information to a specific destination area and transmitted data are encrypted to avoid that attackers can read the content. At the same time, vehicles privacy is preserved by using pseudonyms that replace real identities.

In this paper, we introduce the novel notion of mobility-cast, which has been thoroughly motivated in the previous section, and present a privacy-preserving version of mobility-cast. Differently from previous work such as [14], we consider fine-grained private matching of user attributes (in our case, mobility profiles) as a *building block* of a more general routing protocol, namely, mobility-cast. In addition, our contribution aims at thoroughly analyzing the performance of the proposed mobility-cast protocol, also in comparison with alternative designs. Finally, a proof-of-concept implementation of the protocol on Android smartphones is presented and evaluated.

4. Defining place-friends

In order to define place-friends, we need to formally define a notion of individual mobility pattern, and a similarity metric between mobility patterns. Concerning the definition of mobility pattern, two approaches are typically used in the literature: a point-of-interest based approach, or a partition-based approach. In the former approach, a number of points-of-interest (shopping centers, touristic attractions, public parks, etc.) are identified within the area of interest (typically, a city). A user's mobility profile is then defined by the visiting frequency of the points-of-interest. This notion of mobility profile is used, e.g., in [1]. In the partition-based approach, the area of interest is partitioned into a number of non-overlapping regions, and a user's mobility profile is given by the visiting frequency of each sub-region. Sub-regions typically are defined as the coverage area of a cell-tower (see, e.g., [25]), or based on a square cell partitioning (see, e.g., [4,9,15]).

While in principle our ideas can be applied to any definition of mobility pattern, for the sake of definiteness in the following we use a square grid partition-based approach. More specifically, we assume the mobility region R is a square of side ℓ, which is logically partitioned into $m = h^2$ square cells of side $\frac{\ell}{h}$, where h is a tunable parameter. Assuming an arbitrary order of the m cells, the mobility pattern of a user A is defined as an m-dimensional vector $M_A = (x_1^A, \ldots, x_m^A)$ of real numbers $x_i^A \in [0, 1]$, where x_i^A denotes the relative visiting frequency for the i-th cell, and $\sum_i x_i^A = 1$.

Given the above definition, and in accordance with [9], a user's mobility pattern can be represented as a vector (point) in an m-dimensional vectorial space. Different similarity metrics can be used to compare two mobility patterns. In [9], the authors suggest to use Euclidean distance between the two points corresponding to the individual mobility patterns. Alternatively, one can use the cosine similarity metric used in [8] to quantify similarity between user interests, where interest profiles are represented as points in a vectorial space as well. However, we have to consider that vectors representing mobility patterns are likely to be highly skewed, with most cells visited with near zero frequency, and only a few cells visited on a regular basis. This observation comes from recent studies showing that individuals tend to spend most of the time in a few locations: more specifically, the visitation frequency of locations follows a Zipf's law with exponent 1.2 [25], corresponding to having an individual spending about 60% of the time in the 5 most popular locations. Thus, similarity metrics that consider all coordinates in the vectorial space such as Euclidean distance and cosine metric tend to shallow the relative difference/similarity between mobility patterns, due to the many close-to-zero coordinates which are present in the overwhelming majority of mobility patterns.

To get around this problem, in this paper we use a similarity metric based on comparing the k cells most frequently visited by users. More specifically, let $F_A = \{i_1^A, \ldots, i_k^A\}$ and $F_B = \{i_1^B, \ldots, i_k^B\}$ be the set of most frequently visited cells of user A and B, respectively, where i_j^X denotes the ordinal number in the cell ordering of the j-th most frequently visited cell of user X. We say that users A and

B are *place-friends* if and only if

$$|F_A \cap F_B| \geq \hat{\lambda} \ ,$$

where $\hat{\lambda}$, with $1 \leq \hat{\lambda} \leq k$, is a tunable integer parameter representing the minimal degree of similarity needed to declare two users place-friends. For any user u_i, the set of place-friends of u_i is denoted $S(u_i)$.

Notice that, differently from other metrics such as Euclidean distance and cosine metric, the notion of similarity defined above is apt to a scenario in which most of the x_i values in a mobility profile are near-zero, since only the most frequently visited cells are accounted for in the similarity metric. Furthermore, the notion of similarity defined above is apt to a privacy-preserving implementation, using well-known secure two-party protocols for private set intersection computation (see below).

5. The mobility-cast protocol

Participants in Mobility-Cast are users (also called *agents* in the following) who have a GPS-equipped device, like smartphones or smartwatches, that can keep trace of their mobility pattern during the daily life. In our protocol, we consider that the map of a zone, e.g., a city, is split into cells. Cells can assume different size, for instance they can be represented by a square where each side is long 10, 100 or 1000 m. Clearly, there is a tradeoff between location accuracy (lower with larger cells), and memory requirements on the device (larger with smaller cells).

The current location of a user is collected at regular time intervals (e.g., a few minutes), and it is stored in a file. At regular intervals (say, every few hours or a day), all the collected data are processed to calculate the visiting frequency $f_{new}(C_i)$ for each cell C_i in the map. The new frequency values are combined with the previously stored frequency value $f_{old}(C_i)$ to compute the current mobility profile of the user. We use the typical exponentially weighted moving average (EWMA) to compute the mobility profile of the user, i.e., we update the frequency value $f(C_i)$ for cell C_i as follows:

$$f_{old}(C_i) = f(C_i), \qquad f(C_i) = \alpha f_{new}(C_i) + (1-\alpha) f_{old}(C_i) \ , \quad (1)$$

where $0 < \alpha < 1$ is the degree of weighting decrease.

Starting from the vector $f(C_i)$ of frequency values for each cell C_i, our protocol builds the user mobility profile by maintaining a list of the IDs of the k most visited cells, i.e., the index set F_A for user A. Notice that, since our protocol is based on computing the set intersection between the F sets, elements in F_A are not ordered.

As an example, Fig. 1 shows the mobility region (enclosed by thick lines), which is divided into a set of 9 equal sized cells; *Cell 0, Cell 1*, ..., *Cell 8*. The travel path of an agent u_i in an observation period (say, a day) is indicated by the dashed-dotted line in the figure, and every small black filled circle on the travel path denotes a location at which the position of the mobile device is logged. For each individual agent u_i, a visit frequency f_{new} to every single cell *Cell j* is computed based on the number of logged locations belonging to that cell. Referring again to Fig. 1, the travel path of u_i consists of a set of 18 location logs, denoted ℓ_1 to ℓ_{18}. Hence, the visit frequencies of, for instance, *Cell* 0 and *Cell* 1 are $\frac{5}{18}$ and $\frac{6}{18} = \frac{1}{3}$, respectively. These frequency values are then combined with previous frequency values according to Eq. (1), and used to build the mobility profile.

In order to provide a privacy-preserving comparison between user mobility profiles, we propose a solution that uses the Secure Two-party Computation idea proposed by [26]. While, the specific mechanism used to implement the secure two-party computation of function $g(F_A, F_B)$ is presented in Section 8. We recall that in a secure two-party computation, we have two parties (Alice and Bob), each holding some private data x and y, respectively. The goal of secure two-party function computation is allowing Alice and Bob to jointly compute the outcome of a function $g(x, y)$, without disclosing to the other party the own input. The straightforward way to solve the above problem would be to have a Trusted Third Tarty (TTP) to which Alice and Bob securely send the data, and to have the TTP compute $g(x, y)$ and separately send the outcome to Alice and Bob. The business in secure two-party computation amounts to securely compute $g(x, y)$ without the need of a TTP.

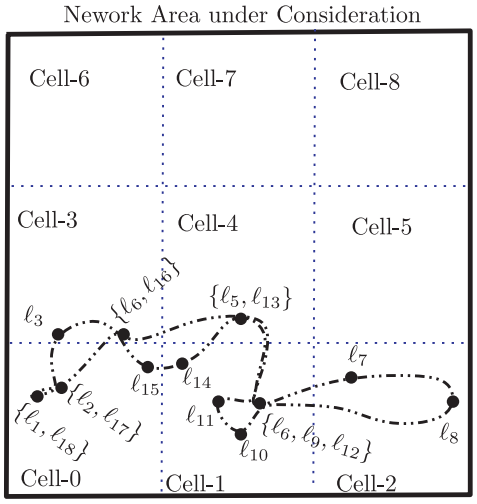


Fig. 1. Building a user's mobility profile. A small black filled circle indicates a location at which the position of the mobile device is logged; multiple names (in braces) to a same location indicate that this location has been logged at different time instants..

In Fig. 2 we pictorially present our protocol, which makes use of Secure-two party computation to compare Alice and Bob mobility profiles. The protocol assumes a threshold value $0 < \lambda \leq \hat{\lambda}$, known to all participants, which is used to control the message forwarding process. The protocols starts when Alice and Bob are in close proximity; for instance, using a Bluetooth connection, when they are less than 20 m apart. Initially, Alice starts a connection to Bob; once received the connection request from Alice, Bob starts the private computation of the function:

$$g(F_A, F_B) = \begin{cases} \textbf{True} & \text{if } |F_A \cap F_B| \geq \lambda \\ \textbf{False} & \text{otherwise} \end{cases} . \quad (2)$$

If the profiles are found to be similar, Alice and Bob are estimated as place-friends, and they start comparing the content of their buffers and exchange files. Otherwise, the connection is terminated. Notice that, if $\lambda < \hat{\lambda}$, function $g(F^A, F^B)$ might evaluate at **True** even if Alice and Bob are *not* place-friends. This situation can be avoided by setting $\lambda = \hat{\lambda}$. However, as we shall see in the next section, increasing the value of λ is detrimental for privacy preservation since more information about the own mobility pattern is disclosed to the other party during the computation. For this reason, using a value of λ lower than $\hat{\lambda}$ is often preferable in practice.

The message forwarding policy is as follows. If Alice is the source of a message M, or if Alice received the message *directly* from the source, M is delivered to Bob if Alice and Bob are estimated to be place-friends according to function (2). In all other cases, including the case in which Alice and Bob are estimated to be place-friends but Alice received M from a user, who is not the source of M, M is not delivered to Bob. Notice that this forwarding policy, which can easily be implemented including a hop-count field in the message, ensures that any message travels at most two-hops to reach a place-friend. For this reason, we name our protocol 2-hops mobility-cast (*MC2H* for short).

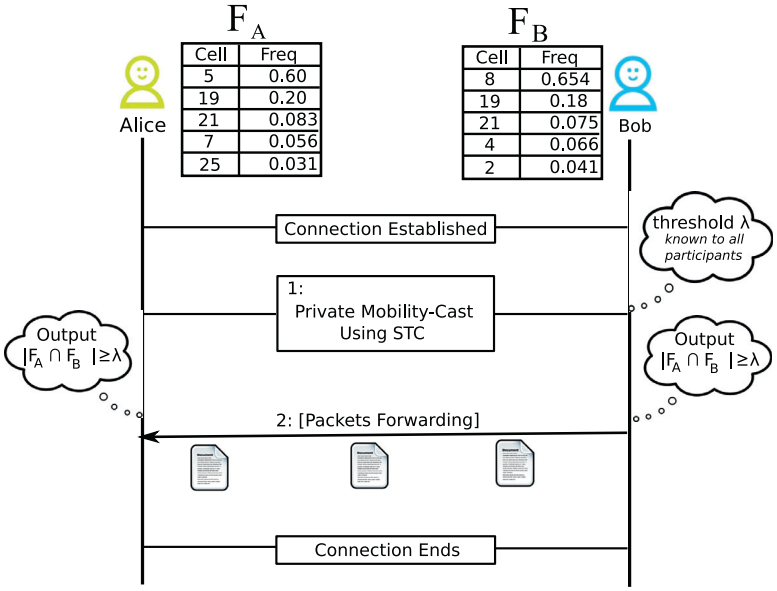


Fig. 2. Protocol flow to discover mobility profiles similarity.

Message forwarding to two hops finds is also motivated by the need of preserving privacy. In fact, Alice and Bob *always use their own mobility profiles to estimate place-friendships*, but they want to achieve place-friendship by keeping their mobility profile private. Hence, the decision on whether a message M generated by a user S and currently in Alice's buffer should be forwarded to Bob is not based on the similarity between Bob's and S's mobility profiles, but on the similarity between Alice's and Bob's profiles. This implies that the message M can be delivered to users that are not place-friends of S, impacting the precision of the forwarding process. It is easy to see that the higher the hop distance from S, the higher the likelihood of forwarding the message to a false place-friend of S. On the other hand, message forwarding is useful to speed up the message propagation process and increase coverage. Restricting forwarding up to the second communication hop is a compromise that has been shown to work well in related work [16].

6. Privacy analysis

While not disclosing user mobility profiles, a certain leakage of private information is unavoidable when using secure two-party computation. In particular, at the end of the protocol computation, the following information is leaked to the other party:

- – if the outcome of $g(F_A, F_B)$ is **True**, the party (say, Bob) knows that at least λ of his most popular locations are in common with Alice. However, he does not know the exact number of common locations (can be any number in the $[\lambda, k]$ interval), nor which they are exactly. Only in the case that $\lambda = k$ Bob knows that Alice has the same exact mobility profile as the own profile.
- –If the outcome of $g(F_A, F_B)$ is **False**, Bob knows only that less than λ of his most popular locations are in common with Alice. However, he does not know the exact number of common locations (can be any number in the $[0, \lambda - 1]$ interval), nor which they are exactly.

To quantitatively evaluate privacy leakage, we use the entropy-based privacy preservation metric introduced in [16]. In particular, we want to quantify the privacy leakage caused by the protocol execution, under the assumption that the attacker's goal is discovering the other party's mobility profile, i.e., his/her k most frequent locations. Taking w.l.o.g. Alice's perspective, Bob's profile is a set of k cell IDs, which can be modeled as a random variable $Y = (y_1, \ldots, y_k)$. Each specific realization of r.v. Y is denoted Y_i, and corresponds to a set of k cell IDs chosen amongst the m possible cell IDs. Hence, the number of possible values of r.v. Y is $\binom{m}{k}$.

The *bit entropy* of a random variable Y with possible values $\{y_1, \ldots, y_n\}$ is defined as [27]:

$$H[Y] = -\sum_{i=1}^{n} p(y_i) \log_2 p(y_i) \ ,$$

where $p(y)$ is the probability mass function of random variable Y. The *privacy preservation* metric of a certain protocol **P** is defined as

$$pp(\mathbf{P}) = \frac{H[Y_{after}]}{H[Y_{initial}]} \ ,$$

where $Y_{initial}$ and Y_{after} are the r.v. modeling Alice's uncertainty about Bob's profile *initially* and *after* the execution of protocol **P**, respectively. The pp metric takes values in [0, 1], with 0 indicating that after **P**'s execution Alice knows exactly Bob's mobility profile (zero privacy preservation), and 1 indicating that after **P**'s execution Alice has the same knowledge about Bob's profile he had before executing the protocol (maximal privacy preservation).

To quantify the pp metric, we need to make same assumptions about the distribution of r.v. Y. In the following, we quantify privacy leakage under the assumption that all locations have the same probability of being included in a node's mobility profile. In other words, we assume that all $\binom{m}{k}$ possible subsets of k out of m possible cell IDs are equiprobable. Notice that this assumption is not necessarily in contrast with the observation made in [25] that people tend to frequently visit only a few locations. In fact, people in general have different more frequently visited location [28], and the resulting aggregate location popularity (which is the one that determines the distribution of r.v. Y) might be relatively uniform. On the other hand, analyzing privacy leakage under a non-uniform location popularity assumption (e.g., assuming Zipf's law) is cumbersome, due to the need of computing each single $p(y_i)$ value in the definition of bit-entropy. This further justifies our working assumption of uniform location popularity.

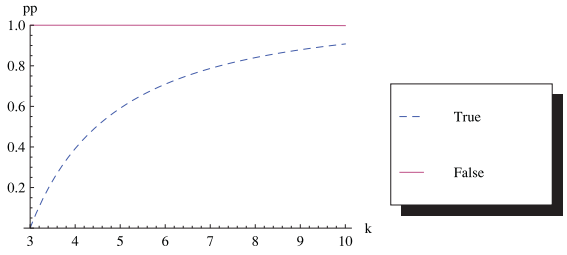


Fig. 3. Value of the *pp* metric for increasing values of k, with parameter λ fixed to 3.

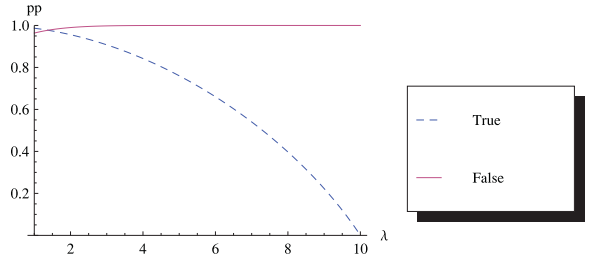


Fig. 4. Value of the *pp* metric for increasing values of λ, with parameter k fixed to 10.

Let us first compute $H[Y_{initial}]$. If locations (cell IDs) have uniform popularity, from Alice's perspective any of the $\binom{m}{k}$ possible Bob's mobility profiles has the same probability $1/\binom{m}{k}$ of occurrence. Hence, we get:

$$H[Y_{initial}] = -\sum_{i=1}^{\binom{m}{k}} p(y_i)\log_2 p(y_i) = -\sum_{i=1}^{\binom{m}{k}} \frac{1}{\binom{m}{k}}\log_2 \frac{1}{\binom{m}{k}} =$$

$$= \sum_{i=1}^{\binom{m}{k}} \frac{1}{\binom{m}{k}}\log_2 \binom{m}{k} = \log_2 \binom{m}{k} .$$

Let us now compute $H[Y_{after}]$. We recall that the value of λ used to estimate place-friendship is fixed and known to both parties. We distinguish the case of protocol execution with outcome **True** or **False**.

If the outcome is **True**, after the protocol execution Alice knows that Bob's mobility profile has at least $\lambda \leq k$ locations in common with the own profile. Fixed a value h, with $\lambda \leq h \leq k$, of possible common locations, the number of possible choices for Bob's profile with h locations in common with Alice can be computed as follows:

$$\binom{k}{h} \cdot \binom{m-k}{k-h} .$$

In fact, the first binomial coefficient accounts for all possible choices of the h locations in common with Alice's profile, taken amongst the k locations in Alice's profile. The second binomial coefficient accounts for all possible choices of the remaining $k-h$ locations in Bob's profile, which are taken amongst the $m-k$ locations which are not in common with Alice's profile.

Given the above, we can compute $H[Y_{after}]$ in case of **True** outcome as follows:

$$H[Y^T_{after}] = \log_2 \left(\sum_{h=\lambda}^{k} \binom{k}{h} \cdot \binom{m-k}{k-h} \right) .$$

If the outcome is **False**, Alice knows that all possible Bob's profiles with at least λ locations in common with the own profile should be excluded from the universe of possible profiles, i.e.,

$$H[Y^F_{after}] = \log_2 \left(\binom{m}{k} - \sum_{h=\lambda}^{k} \binom{k}{h} \cdot \binom{m-k}{k-h} \right) .$$

The value of the *pp* metric for increasing values of k and $\lambda = 3$ is reported in Fig. 3. As expected, a **True** outcome of the protocol's execution discloses more information to the adversary. However, the amount of information disclosed to the other party can be reduced by increasing the value of the number k of locations in the mobility profile. Notice, though, that increasing the value of k beyond a reasonable value has a negative effect on the accuracy of the mobility-cast operation, indicating a tradeoff between networking performance and privacy already observed in [16] for the case of interest-cast.

Fig. 4 reports the *pp* metric for increasing values of λ, with parameter $k = 10$. In case of **True** protocol outcome (the most critical case for privacy leakage), we can reduce privacy leakage by *reducing* the value of λ. Also in this case the need of privacy preservation is at odds with the accuracy of the mobility-cast operation: with a lower value of λ, relatively less locations must be in common to estimate the two parties as place-friends, thus potentially propagating a message M to users who are not actual place-friends of the source of M.

Finally, we briefly analyse the case in which an attacker uses his network interface to eavesdrop the channel trying to infer whether Alice and Bob are place-friends. To this purpose, we split our protocol flow into two phases: i) the *Secure Computation Set Intersection* and ii) the *Packet Forwarding*. During the first phase, the attacker observes only parts of the secure-two party protocol that are not significant for him since he eavesdrops messages masked with the garbled boolean circuits technique [29], and so, he is not able to read sensible data such as highly frequented cells, or common cells. During the second phase, which occurs only if Alice and Bob have common cells, the attacker may see that a file transfer is going on, from which she may infer that Alice and Bob are place-friends. However, the attacker is not able to know their common cells. A solution for this kind of attack would be to use of a cryptographic tunnel between Alice and Bob. In fact, the tunnel will hide the entire communication session between Alice and Bob, and the attacker would not be able to deduce whether they are place-friends.

7. Experiments

7.1. Mobility model

Agent mobility patterns play a key role in determining the performance of a forwarding protocol in opportunistic networks. Therefore, the choice of a proper mobility model from the available pool of models is very important. In this study, we consider a simple yet elegant mobility model [30], which is derived from the analysis of real human mobility traces. Moreover, the model in [30] has incorporated a set of important properties that has been observed in many other trace analyses. We briefly describe the properties that this model has incorporated along with the trace analysis that has observed these properties.

The study in [30] has analyzed two sets of reals traces: *i*) traces of three million mobile phone users where the location of the cell tower that routes user phone calls was recorded for a period of one year, and *ii*) traces of one thousand mobile phone users who registered for a location-based service, where the location of every user was traced every hour for a period of two weeks. The analysis of these traces has unveiled the following properties:

1. the distribution of both jump length[4] (Δ_r) and waiting time[5] (Δ_t) follow a fat-tailed distribution [31–33].
2. the expected number $L(t)$ of distinct locations visited by an individual up to time t follow $L(t) \sim t^{\mu}$, where $\mu = 0.6 \pm 0.02$ [30].
3. the visiting frequency $F(k)$ of the k^{th} most visited location in the trajectory of a human movement follow $F(k) \sim k^{-\alpha}$, where $\alpha = 1.2 \pm 0.1$ [32].
4. the mean squared displacement of the locations of a human movement trajectory follows a slower than logarithmic growth [30].

Interestingly, all these properties can be reproduced in synthetically generated traces using the model introduced in [30], which is also easy to implement and has been used in this paper.

In this model, every agent visits a location (e.g. a Cartesian point in a 2D plane) and waits at that location for a period of time. Movement of an agent is achieved in two steps, and it is controlled by only two parameters, ρ and γ.

1. Once the waiting time at the current location is finished, the users visits a new location (i.e. a location that it has not visited before) with a probability $Pr_{new} = \rho \mathcal{L}^{-\gamma}$, where $\mathcal{L}$ is the number of distinct locations that the user has already visited. In this case, the distance to the new location from the current location is sampled from a power law distribution, and the direction to the new location with respect to current location is chosen randomly from a range $[0, 2\pi]$.
2. With probability $Pr_{old} = 1 - Pr_{new}$, the agent returns to an already visited location. In this case, the location with higher number of previous visits has a higher chance to be visited next.

Essentially, the parameters ρ and γ controls the probability of exploring an unvisited location. As can be understood, the lower the value of ρ, the smaller the chance to visit a new location for a fixed γ, and vice versa. The waiting time at a location is also sampled from a power law distribution.

7.2. Performance metrics

In this work, we are interested in spreading a message M from a source agent u_i to all her place-friends $S(u_i)$, which are the *intended destinations* of M. While we aim at delivering a message M to all its destinations, we would like to have M to be received by the minimum possible number of agents other than the intended destinations. This is desirable for both confidentiality reasons, as well as for reducing the communication overhead in the network. Let us denote by $R(M)$ the set of agents who have received a message M in the process of message delivery from a source u_i to the agents in $S(u_i)$. We consider two primary performance metrics:

Coverage – the ratio between the number of agents who are place-friends of the source and have received the message, and the total number of place-friends of the source, i.e.,

$$Cov = \frac{|R(M) \bigcap S(u_i)|}{|S(u_i)|} \tag{3}$$

Precision – the ratio between the number of agents who are place-friends of the source and have received the message, and the total number of agents who have received the message, i.e.,

$$Prec = \frac{|R(M) \bigcap S(u_i)|}{|R(M)|} \tag{4}$$

Note that a given protocol may perform well considering any one of precision or coverage, while performing poorly with respect to the other metric. Since our goal is to design a protocol that performs well with respect to both metrics, we consider the well-known F-Score metric [34] that combines precision and coverage:

- **F-Score** – the weighted mean of the precision and the coverage, computed as:

$$F - score = 2.\frac{Prec \cdot Cov}{Prec + Cov} .$$

Notice that, since both precision and coverage varies in [0, 1], so does the F-score value, with relatively higher values indicating relatively better performance.

We also consider the following performance metrics:

- **Cost** – the ratio between the number agents who have received a copy of message M including the source u_i, and the total number of agents in the network except the source, i.e.,

$$Cost = \frac{|R(M)|}{n-1},$$

where n is the total number of agents in the network.

- **Delivery Delay** – the average of the delivery delays experienced by those agents who have received the message M, i.e.,

$$Delay = \frac{\sum_{u_j \in R(M)} \delta_j}{|R(M)|} ,$$

where δ_j is the time difference between the instant at which the message M is created and the instant at which agent u_j receives it.

7.3. Simulation setup

Our experiments are executed in three steps. In the first step, we generate a set of mobility traces for every agent using the mobility model described in Section 7.1. In the second step, a subset of agents from the whole set of agents are selected as sources of a set of messages. Finally, a set of message forwarding protocols is run to spread a message from a given source in the network, where the agents move following the mobility traces generated in the first step. The simulation parameters and corresponding values are reported in Table 1; unless stated otherwise, the default values of the parameters are shown in this table.

7.3.1. Generating mobility traces

Recall that the definition of place-friend (Section 5) involves a parameter k, representing the number of most frequently visited locations used to build the mobility profile. In this study, we consider $k = 10$. Therefore, each agent must visit at least 10 distinct cells to have a valid mobility profile, which is achieved both by suitably considering the range of jump-lengths in the mobility model, and by properly tuning the cell size h. We consider $h = 50\ m$, yielding 400 cells in the whole networked area, and this value of h is kept constant in all the simulations performed in this paper. Traces in which the number of distinct cells visited by an agent is less than 10 are discarded. The remaining traces are considered as *valid mobility traces*, and are used to create instances of networks with n agents. The results reported in the following are computed averaging the results over 10 network instances, for each considered parameter setting.

[4] Jump length is defined as the geographic distance between the locations associated with two consecutive sightings of the same user.

[5] Waiting time is defined as the time duration in a location for which a user is continuously sighted.

Table 1
Summary of simulation parameters and their corresponding values.

Parameter description	Value(s)
Simulation area (R^2m^2)	$R = 1000\ m$
Cell size (h^2m^2)	$h = 50\ m$
Number of agents (n)	$n = \{100, 200, \ldots 1000\}$
Radio range (r)	$r = 10\ m$
Velocity (v)	$v = 1\ m/sec$
Jump length (l) [30]	$Pr(l)\ l^{-d},\ d = 1.55,\ 50m \leq l \leq \frac{\sqrt{2}L}{2}$
Waiting time (w) [30]	$Pr(w)\ w^{-e},\ e = 1.8,\ 10s \leq w \leq 100s$
Parameters for defining place-friend	$k = 10, \lambda = \{1, 2, 3, 4, 5\}$
Mobility parameter ρ [30]	$\rho = 0.6$
Mobility parameter γ [30]	$\gamma = -0.21$
Probability p used in 2HPr protocol	$0.01 \leq p \leq 0.12$
Number of simulations	10
Time gap in generating the mobility trace files	3 time steps
Number of messages	20 per simulation

7.3.2. Message forwarding protocols

The performance of *MC2H* is compared to that of three existing protocols, which are described in the following along with their known advantages.

1. *Direct Delivery (DD)*. The source is the only agent who can deliver its message to an intended destination during a direct contact opportunity.
   This protocol is not a forwarding protocol in a strict sense as it does not allow any agent other than the source and the destination(s) to carry a message in the network. Hence, the protocol ensures maximum precision (i.e., $Prec = 1.0$) in any network scenario.
2. *Probabilistic Two-Hop forwarding (2HPr)*. The source creates a message with hop count set to 0. During a contact opportunity, a node carrying a message with hop count ≤ 1 forwards a copy of the message to the encountered node with a fixed probability p, irrespective of whether the encountered agent is a place-friend. In case it is forwarded, the hop count of the duplicated message is increased by 1.
   Restriction of 2 hops forward in this protocol aims at compromising communication overhead with coverage. The value of p is chosen in such a way that the communication overhead of *2HPr* is comparable to that of *MC2H*.
3. *Epidemic Protocol (EP)* [35]. Any agent carrying the message creates a copy and forwards it to any other agent (having no copy of this message) during a contact opportunity, irrespective of whether it is a place-friend, and of the hop count in the message.
   Since the protocol exploits all possible communication opportunities for forwarding the message, it achieves maximum coverage in any network scenario.
4. *Mobility-cast with 2-hops (MC2H)*. The source creates a message with hop count set to 0. During a contact opportunity, an agent carrying a message with hop count ≤ 1 creates a copy of the message and forwards it only if the encountered node is a place-friend. In case it is forwarded, the hop count of the duplicated message is increased by 1.
   The protocol aims at achieving a good compromise between precision and coverage (i.e., high F-score), while trying to reduce the communication overhead.

7.3.3. Generating a message

In principle, any agent can generate a message and be a source. However, relevant to our study are only those cases in which a source has a positive number of place-friends to which the message should be delivered. For this reason, we select message sources as follows. Initially, based on the mobility traces we find the set $S(u_i)$ of place-friends for every agent u_i in a network. Then, each of those agents for which $|S(u_i)|$ is above a threshold is considered as a potential message source when messages are randomly created. Notice that, in order to guarantee a fair comparison between the various forwarding protocols considered, once a message source and generation time are randomly selected, the same source and generation time are used for all the compared protocols.

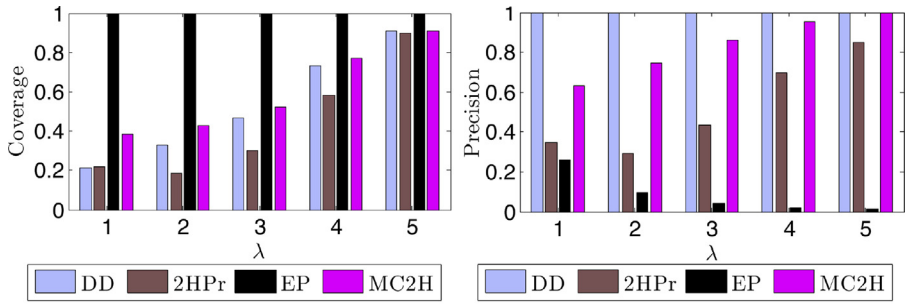
7.4. Results

In this section, we analyze the performance of the protocols mainly by varying two parameters: the parameter λ used to defined place-friend; and the number n of agents in a network, in order to investigate the scalability of our proposed protocol.

7.4.1. Impact of λ

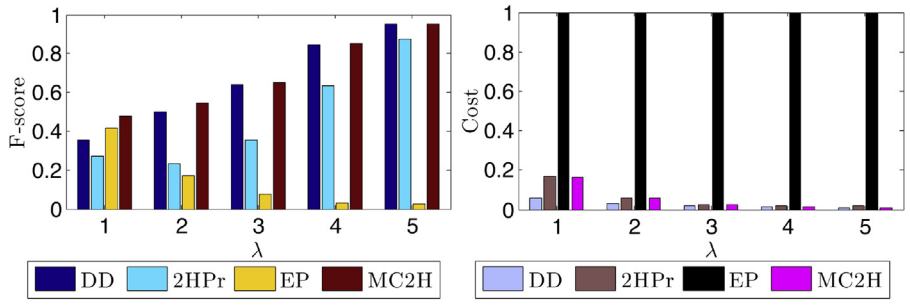
All the results in this subsection are obtained considering $n = 100$ agents in a network. Fig. 5a shows the variation of coverage with parameter λ. As expected, the coverage in case of epidemic protocol (EP) is the highest (i.e., 1.0) irrespective of λ, because this protocol spreads a message to all the agents in a network. The coverage in case of 2HPr is almost equal to that of direct delivery (DD) protocol when $\lambda = 1$, and it is minimum for all other values of λ. This is due to the fact that DD always deliver the message to a source's place-friend when there is a direct contact opportunity, while in case of 2HPr message forwarding is probabilistic, and oblivious to place-friendship. The coverage with MC2H protocol is about 80% higher than that in DD when $\lambda = 1$. However, the improvement in coverage using MC2H reduces as λ increases, and the coverage becomes almost the same as DD when $\lambda = 5$. This is likely due to the variation in the number of place-friends of any given agent, which becomes smaller as λ increases. Moreover, defining place-friends using a higher values of λ, e.g., $\lambda = 5$, imposes a stronger similarity between mobility profiles of place-friends, hinting to the fact that the source is more likely to directly meet its place-friends.

Fig. 5b shows the variation of precision with λ. As expected, the precision in case of DD is always at maximum (i.e., 1.0) irrespective of λ. On the other hand, the precision in case of EP is always at the lowest irrespective of λ, and it is decreasing as λ increases. Note that the precision in case of EP essentially indicates the fraction of agents that are place-friends of a given agent on an average. The number of place-friends is about 27 agents when $\lambda = 1$, while it is in between 1 or 2 agents when $\lambda = 5$. Interesting to observe here is that the precision in case of MC2H slowly increases with λ, and it becomes almost equal to that achieved by DD when $\lambda = 5$. The precision in case of 2HPr lies somewhere in between that of DD and MC2H.



a Variation of coverage with parameter λ.

b Variation of precision with parameter λ.



c Variation of F-Score with parameter λ.

d Variation of cost with parameter λ.

Fig. 5. Variation of cost with parameter λ.

Fig. 5c reports the F-score of the various protocols. Results show that the F-score in case of EP reduces as λ increases, which is mainly due to an increasingly lower precision with increasing values of λ, as compared to that of other protocols. On the other hand, the good F-score achieved by DD is mainly due to its maximal precision. Interesting to observe that the F-score performance of MC2H is the best when $\lambda < 5$, and it is slightly lower than that in DD when $\lambda = 5$. This is because of slightly lower precision in MC2H compared to that in DD, and because DD achieves a good coverage when $\lambda = 5$ for the reasons explained above. We further investigate this phenomenon in the following.

We also analyze the performance of the protocols in terms of the cost (Fig. 5) and the delivery delay (Fig. 6a). The cost in EP is always the highest, as EP enforces the maximal number of agents in a network to receive a message. It is the lowest in DD, as the protocol restricts a message from being delivered to agents which are not place-friends of a source. In MC2H, the cost is slightly higher than that in DD when $\lambda \leq 3$, however, it becomes almost same as that of DD when λ is above 3. This result is interesting as our proposed protocol achieves a better F-score (Fig. 5c) without a significant increase in the cost when $\lambda \leq 3$. Moreover, the costs in 2HPr and in MC2H are kept almost similar by tuning the probability p in 2HPr[6]; these value pairs are kept unchanged for all subsequent experiments where 2HPr is considered. This essentially suggests that the number of agents who can receive a message in a network is almost same using any of the two protocols. However, F-score result (in Fig. 5c) shows that the performance in MC2H protocol is significantly better than 2HPr, indicating the better effectiveness of MC2H forwarding mechanism.

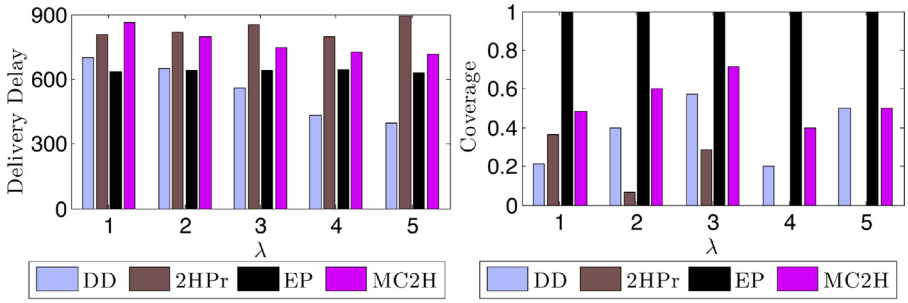
Analyzing the delivery delay, we see (in Fig. 6a) that the delay is minimum in case of EP when $\lambda < 3$. However, the delay in EP becomes more than that in DD when $\lambda \geq 3$. This is because the time it takes for a message to reach only a fraction of place-friends of a source in DD becomes less than that to reach all the agents in the network in EP when $\lambda \geq 3$. The delay in MC2H is higher than that in DD when $\lambda \leq 3$, however it becomes almost double the delay in DD when $\lambda > 3$. Thus, we see that there exist a threshold on λ, which is essentially a threshold on the number of place-friends, beyond which MC2H may not perform reasonably better than DD in terms of the delivery delay. We investigate the issue of characterizing this threshold in the following.

We have seen the performance of MC2H is no better than DD in some cases, in particular, when $\lambda > 3$. We have anticipated that this is likely because the number of place-friends of a source becomes less than a threshold, and the source itself meets almost all of its place-friends. To further investigate this, we have considered only one message to be generated in a network from a source which has the highest number of place-friends for each setting of $\lambda = 1$ through 5. We can see that the coverage in case of MC2H is much higher than that of DD up to $\lambda = 4$ (Fig. 6b). Also, the precision in case of MC2H becomes as high as DD when $\lambda \geq 3$ (Fig. 6c). Thus, we can see that our proposed protocols MC2H performs better than DD when the number of place-friends is larger than a minimum value[7], a situation in which delivering messages to place-friends becomes non-trivial since most place-friends cannot be directly met by the source.

Fig. 6d shows the F-score observed in all the protocols. Interesting to see that the F-score is significantly higher in case of MC2H

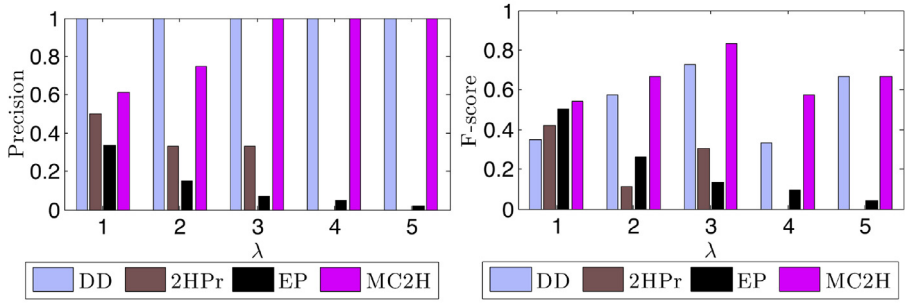
[6] Values of p are set as follows: $p = 0.12, 0.05, 0.02, 0.01$ and 0.01 when $\lambda = 1, 2, 3, 4,$ and 5, respectively.

[7] This value corresponds to about 5% of the total number of agents, as it can be deduced by the fact that the precision in EP is about 0.05 when $\lambda = 4$.



a Variation of average delivery delay with parameter λ.

b Variation of coverage with λ; only one message is generated from a source having maximum number of place-friends.



c Variation of precision with λ; only one message is generated from a source having maximum number of place-friends.

d Variation of F-score with λ; only one message is generated from a source having maximum number of place-friends.

Fig. 6. Variation of average delivery, coverage, precision and F-score.

up to $\lambda = 4$ than the other protocols considered. The F-score in case of 2HPr cannot be observed (i.e., 0) when $\lambda > 3$ as it does not cover any agent who is a place-friend of the source in this particular experiment.

7.4.2. Impact of n

In this subsection, we investigate the performance of the protocols as a function of the number of agents n in a network. We have investigated the performance with λ varying from 1 to 5, however, we present the results with $\lambda = 1$ because they are the most significant and also for space limitations.

Fig. 7a shows the variation of coverage with n for all the protocols. The coverage increases from 40% to 80% in case of MC2H when n increases from 100 to 1000. The coverage in case of DD remains almost constant at about 20%. As expected, the coverage in case of EP remains at maximum value. In case of 2HPr, it is increasing with n at a rate as that of MC2H, however, the coverage remains smaller than MC2H irrespective of n.

The precision in case of DD remains at maximum, and it is minimum in case of EP (Fig. 7b). The precision in case of MC2H is reduced from 63% to 50% when n is changed from 100 to 300, and thereafter the rate of reduction in the precision is very slow. This is because as the number of place-friends increases, the source cannot meet with them directly. Therefore, MC2H helps to spread a message from a source to those unseen place-friends from only its place-friends with a high probability.

The F-score becomes significantly higher in MC2H than that in any other protocols considered (Fig. 7c). Hence, it is interesting to observe that our proposed protocol can scale well even when a large number of place-friends is present in the network.

7.4.3. Summary of analysis

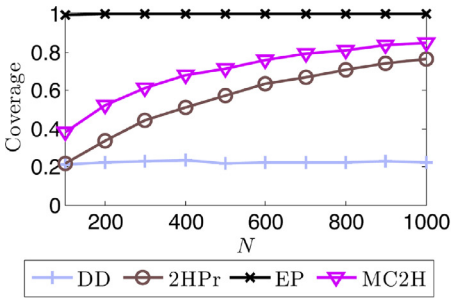
The main findings of our performance analysis can be summarized as follows:

- MC2H provides a relatively good coverage in scenarios where the number of place-friends is more than actual friends, i.e., the place-friends that come in direct contact of a source of a message.
- MC2H performs well in an environment where a message has to be confined only within a limited number of trusted agents, i.e., MC2H offers a significantly higher precision than protocols with similar or higher coverage.
- MC2H performs no worse than DD even when the number of place-friends is very limited.
- MC2H can scale well with the number of agents in a network in terms of both the precision and the coverage, while it keeps the cost as low as in case of DD.

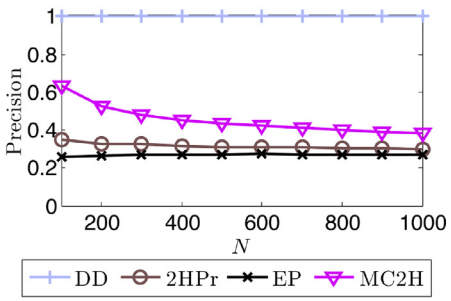
The superior performance of MC2H over competing protocols has been assessed also based on a real-world mobility trace of 77 users. For details, the interested reader is referred to [2].

7.5. Impact of agents with fake profile

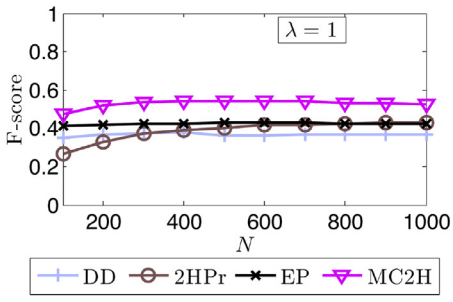
In this section, we analyze the impact of having agents with fake mobility profiles on the performance of the proposed protocol.



a Variation of coverage with number of agents n when $\lambda = 1$ is used to define place-friend.



b Variation of precision with number of agents n when $\lambda = 1$ is used to define place-friend.



c Variation of F-score with number of agents n when $\lambda = 1$ is used to define place-friend.

Fig. 7. Variation of coverage, precision and F-score.

We describe the model of agents with fake profiles, and evaluate the performance of two protocols (DD and MC2H) in the context of this study, considering the other two protocols (EP, 2HPr) to be irrelevant as their performance do not depend on mobility profile.

7.5.1. Attacker model

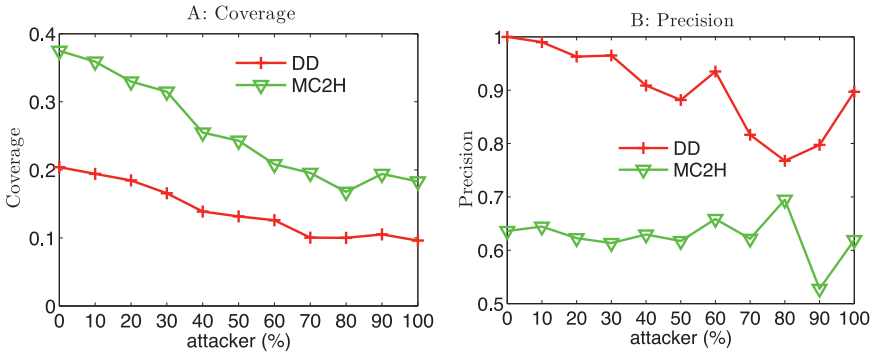
There can be many ways that an agent can alter its mobility profile (lets call them as attacker), ranging from targeting some particular agents to target the entire network. In one extreme, target can be one particular agent, in which case detailed observation of the locations that the target agent moves through is required. In this case, any message targeted to or originating from the targeted agent will be delivered to the attacker. Such an attack model may seem to be impractical as it involves a lot of effort to be invested by an attacker to make the attack successful, but it is not impossible. On the other extreme, target can be the entire network, which imply that an attacker needs to make all possible places in a network to be the frequented places for itself, and put them in the mobility profile with equal probability. Such an attack can essentially cause any message in the network to be delivered to the attacker. However, this type of attack needs a lot of resources to be invested starting from making larger profile, to allocating larger buffer space for the messages to be stored and processed. Hence, this strategy may also be impractical, but again not impossible.

We assume a simple and conceivable strategy for an attacker and easy to launch. An attacker in this case limits the number of places that can be put in its fake profile, and make all these places to be visited with equal probability. Such an attack can form the basis of more targeted attack but neither requires a lot of effort nor a lot of resource as in the extreme cases (discussed above). In this study, we consider 10 distinct places to be present in a fake mobility profile of an attacker with $1.0/10 = 0.1$ as the probability to visit to each of these places. The protocol parameter λ is fixed at 1 in this case, to achieve maximum benefit (in terms of profile match) out of this attack model. We randomly choose a set of agents to be attacker, excluding those agents who can generate a message (i.e., no source agent in the network can become an attacker which ensures that the messages to be legitimate ones, though it need not be the case in the all practical scenarios).

7.5.2. Evaluation of protocol performance

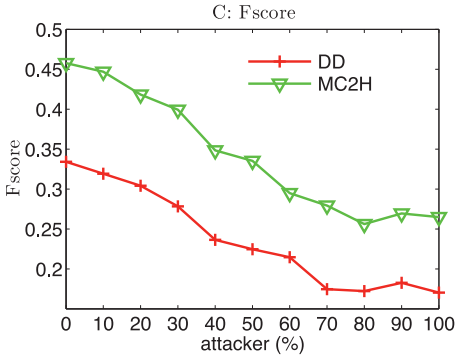
We consider two of our protocols, DD and MC2H for performance evaluation, as the performance of these protocols can vary drastically depending on how many attackers can receive a message. Note that we have the place-friends computed by considering only the true mobility profiles. This can make an actual place-friend of a source agent to turn into an attacker if it has decided to altered its mobility profile (as per the attack model discussed above). Because whether an agent has altered its mobility profile is not known to any other agent, an actual place-friend may not receive a message in an opportunity if it has altered its profile, hence, affecting the performance of a protocol.

Fig. 8a shows the impact of having a number of attackers (i.e., x-axis shows the fraction of agents become attacker) in a network on the coverage of both DD and MC2H protocols (0% attacker essentially indicates that there is no agent with fake mobility profile in the network). As expected, the coverage in both the protocols reduces as number of attackers increases in the network. The rate



a Variation of coverage with number of fake agents when $\lambda = 1$ is used to define place-friend.

b Variation of precision with number of fake agents when $\lambda = 1$ is used to define place-friend.



c Variation of F-score with number of fake agents when $\lambda = 1$ is used to define place-friend.

Fig. 8. Variation of coverage, precision and F-score with fake agents.

at which the coverage reduces with number of attackers is relatively higher in MC2H, which is basically because some of legitimate place-friends might have become attacker and no more part of the valid place-friends in the network.

Fig. 8b shows the impact on the precision. Interestingly, the precision in case of MC2H, though reduces, remain almost similar over the increase of attackers. The precision in case of DD drops significantly compared to MC2H. In case of MC2H, in the first hop, the profile of a source agent has to match with attacker (which is same as the case in DD), but in the second hop the attackers profile may match with the profile of a legitimate place-friend in order to make the message travel though this hop. Therefore, even if a message is passed on to an attacker in the first hop, the attacker in turn can fail to pass on the message though the second hop. Hence, even though the coverage drops rapidly, the precision is not compromised much in case of MC2H.

Finally, the F-score is shown in Fig. 8c. The results show that the F-score drastically reduces with number of attacker (almost linearly degraded) in both the protocols, from 0.45 to 0.26 in case of MC2H, and from 0.34 to 0.10 in case of DD. Interesting to see that the F-score remain at higher level in case of MC2H compared to DD irrespective of the number of attackers in the network. This is essentially due to (more or less) constant precision achieved in MC2H and higher coverage. Overall, the performance of MC2H observed in our experiments holds a level of promise to be utilized in realistic scenarios even when a number of attackers present in a network.

8. Prototype implementation of *MC2H*

In this section, we present two privacy-preserving implementations of MC2H for Android smartphones based on two Secure Two-party Computation (2PC) solutions.

Over the last ten years, researchers have proposed different 2PC frameworks to run private functions. FairPlay [36] is a well-know framework that allows users to write functions using a high level language, called SFDL, and to compile SFDL functions into garbled boolean circuits, which will mask the real inputs of both participants. Only a limited number of commands and operations are available in SFDL. For instance, it is not possible to use text values in a function, but only integers or simple types are allowed.

FairPlay has strong security properties in the context of two-party computation. The framework is shown to be secure against a malicious party; in particular *i*) a malicious party cannot learn more information about the other party's input than it can learn from a TTP that computes the function; and *ii*) a malicious party cannot change the output of the computed function. New versions of this framework are FairplayMP [37], which is the extension of

Fairplay that works with more than two parties, and MobileFairplay [17], which is the version of Fairplay ported to Android Smartphones.

More recent 2PC frameworks are: MightBeEvil [38] and CBMC-GC [3]. Both have the similar goal, namely allows people to easily write functions that can be run in a private way. CBMC-GC is composed of two main parts: the compiler that translates functions written in "C" into garbled circuits, and the interpreter is able to execute compiled functions [39]. Thus, CBMC-GC offers a very flexible high level language that allows developers to express a wider ranges of functions compared to simpler techniques, which for instance only focuses on simple private matching operations. Moreover, CBMC-GC implements an optimization phase during the compilations phase that allows the framework to use less memory of other 2PC frameworks.

8.1. Attacker model

The authors of CBMC-GC and Fairplay claim that their frameworks provide security in the *honest-but-curious* attacker model. Here, the attacker follows all protocol steps as per specifications, but he can try to learn additional information about the other party during message exchanging phase, with the purpose to acquire at least part of the private profile. Moreover, as customary in secure two-party computation, there is an asymmetry on the provided security guarantees. In fact, there is no way to prevent one party from terminating the protocol prematurely, and not sending the outcome of the computation to other party. This situation can be noticed by the weak party, but cannot be recovered from.

In our attacker model, we envision attackers that are not interested in the content of the data forwarded by users. Our MC2H protocol does not protect the *confidentiality* of packets, since this is *not* its design goal. MC2H is designed to exchange messages with users who have similar mobility patterns without revealing those patterns (thus aiming at the privacy of the mobility patterns rather than of the packet content), and this comparison does not use the packet content. As a matter of fact, confidentiality and integrity of packets are orthogonal problems to the one considered in this work, and they could be achieved by means of other techniques.

Finally, we consider that attackers do not modify our implementation to generate false mobility trajectories to execute the "lying attack", in which he is able to send a fake vector of locations that he did not visit. In fact, we suppose that our application automatically establishes the most visited places and sends them during the private matching without any human interactions. The "lying attack" would require a tampering of our application, and its integrity is out of our scope for the moment.

8.2. Prototype performance

The function that we wrote to privately compute similarity between *Alice*'s and *Bob*'s mobility profiles works by comparing each cell ID in *Alice*'s profile with those in *Bob*'s profile. If the same cell ID appears in both profiles, then a counter is increased. At the end of the function, the value of the counter (common frequently visited cells) is compared to the threshold λ known to both parties. If the comparison is positive, then the output for both Alice and Bob is **True** (represented by integer 1), otherwise it is **False** (represented by integer 0).

We wrote the above function using the SFDL language, and C. This permitted us to have two different prototypes to run 2TC private functions and to evaluate their performances running our function. The resulting boolean garbled circuits were integrated into two different 2PC running environments: MobileFairplay, and a novel porting of CBMC-GC to the Android platform that we did to run MC2H within this recent and very efficient framework.

Table 2
Smartphones used for testing computation of function $g(F^A, F^B)$.

Smartphone	CPU	RAM	Android O.S.
Samsung Galaxy S2	Dual-core 1228 MHz	1024 MB	2.3.6
Samsung Galaxy S-Plus	Single-core 1443 MHz	512 MB	2.3.5
Samsung Galaxy Nexus	Dual-core 1200 MHz	1024 MB	4.3
Sony Xperia T	Dual-core 1500 MHz	1024 MB	4.0
LG Nexus 4	Quad-core 1500 MHz	2000 MB	4.4.4
LG-P990	Dual-core 1000 MHz	512 MB	2.3.4

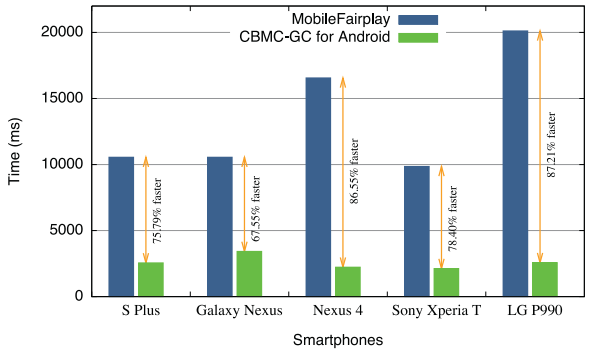


Fig. 9. Times to compute function $g(F^A, F^B)$.

We developed two distinct prototypes of our MC2H protocol, including private estimation of place-friendship. We assume that the user location is logged every minute. Once per day, the application takes the recorded GPS coordinates and calculates visitation frequencies per cells. When two users – *Alice* and *Bob* – meet each other, *Alice* challenges *Bob* on the number of common cells they have in their profiles in a privacy-preserving manner. The current version of both applications considers mobility profiles composed of the five most frequently visited cells, i.e., $k = 5$. *Alice* starts the connection with *Bob* through the bluetooth interface if they use MobileFairplay, or the Wi-Fi interface in case of CBMC-GC running environment[8], and then the Secure-Two party computation function $g(F_A, F_B)$ begins. When the function ends, both users know the value of $g(F_A, F_B)$. Inputs of both participants cannot be manually inserted, but they are directly established and integrated by our application into the 2PC procedure. We consider this to avoid that *Alice* and *Bob* may manipulate their input guessing the same vector input of the other. Finally, if Alice and Bob recognise each other as place-friends, they start an interaction phase consisting in sharing files of small dimensions (`txt`, `pdf`, `jpg`, etc. $\leq$ 10MBytes).

In the following, we report the running time that the $g(F^A, F^B)$ function requires when it is executed first with MobileFairplay and then with CBMC-GC. Table 2 lists the smartphones that we used for our tests. All tests were performed using the Samsung Galaxy S2 device as server, while the others phones connect to it in a client mode.

Fig. 9 shows that the porting of CBMC-GC into the Android platform significantly outperforms MobileFairPlay in the execution of $g(F_A, F_B)$. In fact, the "Sony Xperia T" achieves the best performance using CBMC-GC as 2PC framework, and its running time (2.1 s) is reduced of 78.40% with respect to MobileFairPlay. The second best performing phone is the "Nexus4" (2.2s) , in which case the relative performance increase of CBMC-GC vs. MobilieFairPlay is 86,55%. Similar results are observed for the other smartphones. On average, the reduction in running time achieved by CBMC-GC in comparison with MobileFairplay is about 79%. Finally, we calculated the scalability of CBMC-GC increasing the value of k

[8] This choice comes from the different framework implementations and the impact to the total computational time is negligible.

up to $k = 50$. We found that $k = 50$ is the practical limit of CBMC-GC having our function correctly executed with a running time of 4 *s* (average).

9. Conclusion

In this paper, we have motivated the need of mobility-casting a message in an opportunistic network, i.e., delivering a message to users with a mobility pattern similar to the one of the message source. In fact, such place-friends are likely to include not only current, but also forthcoming social acquaintances of a user. We have then introduced a privacy-preserving design and implementation of mobility cast. By extensive analysis and simulation based on a realistic mobility model, we have verified that the designed protocol is very effective in delivering the message to the intended destinations in situations where the message source does not come into direct contact with all its place-friends, i.e., exactly in those situations where a forwarding protocol is needed. Finally, we have developed two alternative implementations of mobility-cast on Androids platform, and tested them on different smartphones.

As possible avenues for future work, we plan to investigate the interplay between cell size, number of place-friends, and mobility-cast performance, with the of goal identifying the best spatial resolution to be used when building a user's mobility profile.

References

[1] S. Scellato, A. Noulas, C. Mascolo, Exploiting place features in link prediction on location-based social networks, in: Proceeedings of the KDD, ACM, USA, 2011, pp. 1046–1054, doi:10.1145/2020408.2020575.

[2] G. Costantino, F. Martinelli, P. Santi, Privacy-preserving mobility-casting in opportunistic networks, in: PST Conference, Canada, 2014, pp. 10–18, doi:10.1109/PST.2014.6890918.

[3] A. Holzer, M. Franz, S. Katzenbeisser, H. Veith, Secure two-party computations in ansi c, in: Proceedings of the CCS, in: 2012, ACM, 2012, pp. 772–783, doi:10.1145/2382196.2382278.

[4] D.J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, J. Kleinberg, Inferring social ties from geographic coincidences 107 (52) (2010) 22436–22441.

[5] D. Wang, D. Pedreschi, C. Song, F. Giannotti, A.-L. Barabasi, Human mobility, social ties, and link prediction, in: Proceedings of the KDD, ACM, New York, NY, USA, 2011, pp. 1100–1108, doi:10.1145/2020408.2020581.

[6] E.M. Daly, M. Haahr, Social network analysis for routing in disconnected delay-tolerant manets, in: Proceedings of the MobiHoc, in: 2007, ACM, New York, NY, USA, 2007, pp. 32–40, doi:10.1145/1288107.1288113.

[7] P. Hui, J. Crowcroft, E. Yoneki, Bubble rap: Social-based forwarding in delay tolerant networks, in: Proceedings of the MobiHoc, ACM, USA, 2008, pp. 241–250, doi:10.1145/1374618.1374652.

[8] A. Mei, G. Morabito, P. Santi, J. STEFA, Social-aware stateless routing in pocket switched networks, IEEE Trans. Parallel Distrib. Syst. 99 (2014) 1. http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2307857.

[9] J. Leguay, T. Friedman, V. Conan, Evaluating mobility pattern space routing for dtns, in: Proceedings of the INFOCOM 2006., 2006, pp. 1–10, doi:10.1109/INFOCOM.2006.299.

[10] W.J. Hsu, D. Dutta, A. Helmy, Profile-cast: behavior-aware mobile networking, in: Proceedings of IEEE Wireless Communications and Networking Conference(WCNC), Las Vegas, USA, 2008, pp. 3033–3038.

[11] W.J. Hsu, D. Dutta, A. Helmy, Extended abstract: mining behavioral groups in large wireless lans, in: Proceedings of ACM International Conference on Mobile Computing and Networking (MOBICOM), Montréal, Canada, 2007, pp. 338–341.

[12] W. Dong, V. Dave, L. Qiu, Y. Zhang, Secure friend discovery in mobile social networks, in: Proceedings of IEEE Infocom, 2011, pp. 1647–1655.

[13] M. Li, N. Cao, S. Yu, W. Lou, Findu: privacy-preserving personal profile matching in mobile social networks, in: INFOCOM, 2011 Proceedings IEEE, 2011, pp. 2435–2443, doi:10.1109/INFCOM.2011.5935065.

[14] R. Zhang, Y. Zhang, J. Sun, G. Yan, Fine-grained private matching for proximity-based mobile social networking, in: INFOCOM'12, 2012, pp. 1969–1977.

[15] A. Aviv, M. Sherr, M. Blaze, J. Smith, Privacy-aware message exchanges for geographically routed human movement networks, in: S. Foresti, M. Yung, F. Martinelli (Eds.), ESORICS 2012, LNCS, 7459, Springer, 2012, pp. 181–198, doi:10.1007/978-3-642-33167-1_11.

[16] G. Costantino, F. Martinelli, P. Santi, Investigating the privacy versus forwarding accuracy tradeoff in opportunisticinterest-casting, IEEE Trans. Mob. Comput. 13 (4) (2014) 824–837, doi:10.1109/TMC.2013.20.

[17] G. Costantino, F. Martinelli, P. Santi, D. Amoruso, An implementation of secure two-party computation for smartphones with application to privacy-preserving interest-cast, in: Proceedings of Mobicom, in: 2012, ACM, 2012, pp. 447–450, doi:10.1145/2348543.2348607.

[18] M. Mahmoud, S. Taha, J. Misic, X. Shen, Lightweight privacy-preserving and secure communication protocol for hybrid ad hoc wireless networks, IEEE Trans. Parallel Distrib. Syst. 25 (8) (2014) 2077–2090, doi:10.1109/TPDS.2013.298.

[19] M. Li, S. Yu, N. Cao, W. Lou, Privacy-preserving distributed profile matching in proximity-based mobile social networks, IEEE Trans. Wireless Commun. 12 (5) (2013) 2024–2033.

[20] X. Liang, X. Li, K. Zhang, R. Lu, X. Lin, X.S. Shen, Fully anonymous profile matching in mobile social networks, IEEE J. Sel. Areas Commun. 31 (9) (2013) 641–655.

[21] R. Zhang, J. Zhang, Y. Zhang, J. Sun, G. Yan, Privacy-preserving profile matching for proximity-based mobile social networking, IEEE J. Sel. Areas Commun. 31 (9) (2013) 656–668.

[22] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: Proceedings of the International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT)), Prague, Czech Republic, 1999, pp. 223–238.

[23] J.S. Lei Zhang, J. Pan, Towards privacy-preserving and secure opportunistic routings in vanets, in: In Proceedings of IEEE SECON, 2014.

[24] A. Prado, S. Ruj, A. Nayak, Enhanced privacy and reliability for secure geocasting in vanet, in: Communications (ICC), 2013 IEEE International Conference on, 2013, pp. 1599–1603, doi:10.1109/ICC.2013.6654743.

[25] C. Song, Z. Qu, N. Blumm, A.-L. Barabási, Limits of predictability in human mobility, Science 327 (5968) (2010) 1018–1021, doi:10.1126/science.1177170.

[26] C. Andrew, C. Yao, Protocols for secure computations, in: 23rd IEEE Symposium on FOCS, 1982, pp. 160–164.

[27] C. Shannon, A mathematical theory of computation, Bell Syst. Tech J. 27 (1948) 623–656.

[28] Y.-A. de Montjoye, C.A. Hidalgo, M. Verleysen, V.D. Blondel, Unique in the crowd: the privacy bounds of human mobility, Sci. Rep. 3 (2013).

[29] A.C. Yao, Protocols for secure computations, in: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, in: SFCS '82, IEEE Computer Society, Washington, DC, USA, 1982, pp. 160–164, doi:10.1109/SFCS.1982.88.

[30] C. Song, T. Koren, P. Wang, A.-L. Barabasi, Modelling the scaling properties of human mobility, Nat. Phys. 6 (10) (2010) 818–823.

[31] D. Brockmann, L. Hufnagel, T. Geisel, The scaling laws of human travel, Nature 439 (2006) 462–465.

[32] M.C. Gonzalez, C.A. Hidalgo, A.-L. Barabasi, Understanding individual human mobility patterns, Nature 453 (2008) 779–782.

[33] K. Lee, S. Hong, S.J. Kim, I. Rhee, S. Chong, Slaw: self-similar least-action human walk, IEEE/ACM Trans. Networking 20 (2) (2012) 515–529.

[34] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, ACM Press - Addison-Wesley, New York, 1999.

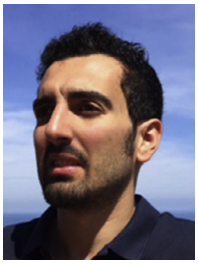
[35] A. Vahdat, D. Becker, Epidemic Routing for Partially Connected Ad hoc Networks, Tech Rep, Duke University, 2000.

[36] D. Malkhi, N. Nisan, B. Pinkas, Y. Sella, Fairplay—a secure two-party computation system, in: Proceedings of the Conference on USENIX Security Symposium, in: SSYM'04, USENIX Association, USA, 2004. 20–20

[37] A. Ben-David, N. Nisan, B. Pinkas, Fairplaymp: a system for secure multi-party computation, in: Proceedings of the 15th ACM CCS, in: CCS '08, ACM, New York, NY, USA, 2008, pp. 257–266, doi:10.1145/1455770.1455804.

[38] Y. Huang, P. Chapman, D. Evans, Privacy-preserving applications on smartphones, in: Proceedings of tjhe USENIX Conference on Hot Topics in Security, in: HotSec'11, 2011. 4–4

[39] V. Kolesnikov, T. Schneider, Improved garbled circuit: free xor gates and applications, in: L. Aceto, I. Damgård, L. Goldberg, M. Halldórsson, A. Ingólfsdóttir, I. Walukiewicz (Eds.), Automata, Languages and Programming, LNCS, 5126, Springer, 2008, pp. 486–498, doi:10.1007/978-3-540-70583-3_40.



Gianpiero Costantino is a Researcher at the Italian National Research Council (CNR). He has been working for the Security group within the Institute of Informatics and Telematics located in Pisa. From November 2007 to March 2011 he was a Ph.D. student at University of Catania and he conducted his research on Trust, Reputation and Power-Saving within Mobile Ad hoc networks. From 2011 to August 2015 he was a Post-Doc Researcher and his research focused on Security and Privacy aspects within Opportunist Networks, Internet of Things, Social Networking. Currently, Dr. Costantino is involved in the CoCoCloud — FP7 Projects —, HC@WORKS-2 — EIT ICT Labs project —, Securing Smart Airport — Enisa Project —, and his research covers privacy aspects on the Cloud using cryptographic techniques, and Trust solutions for automotive.

Rajib Maiti is a Marie Curie Fellow for post-doctoral research at IIT-CNR, Pisa, Italy, with ERCIM fellowship since May, 2014 under the supervision of Dr. Paolo santi. He has completed his PhD in April, 2014, in the area of future internet at the department of computer science and engineering, Indian Institute of Technology Kharagpur, India. Currently, his area of research is centered around protocol development, performance evaluation, and human mobility analysis. He has published several papers in conferences like COMSNETS, ICDCN, INFOCOM, and a paper in journal of advances in complex systems.

Fabio Martinelli received his PhD in Computer Science from the University of Siena in 1999. He is a senior researcher of IIT-CNR. He is co-author of more than 100 scientific papers, and his research interests range from formal methods, distributed systems, computer security and foundations of security and trust. He founded and chaired the WG on security and trust management (STM) of the European Research consortium in Informatics and Mathematics (ERCIM), and he is involved in several Steering Committees of international WGs and/or Conferences/workshops.

Paolo Santi is a Research Scientist at MIT Senseable City Lab where he leads the MIT-Fraunhofer Ambient Mobility initiative, and he is a Senior Researcher at the Istituto di Informatica e Telematica del CNR. His research interest is in the modeling and analysis of complex systems ranging from wireless multi hop networks to sensor and vehicular networks and, more recently, smart mobility and intelligent transportation systems. In these fields, he has contributed more than 100 scientific papers and two books. Dr Santi has been involved in the technical and organizing committee of several conferences in the field, and he is/has been an Associate Editor of the IEEE TMC, the IEEE TPDS, Computer Networks and Guest Editor of the Proceeding of the IEEE.