Computer Communications 000 (2015) 1–8

Contents lists available at ScienceDirect

Computer Communications

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

Lightweight private proximity testing for geospatial social networks

Panayiotis Kotzanikolaou[a], Constantinos Patsakis[a,*], Emmanouil Magkos[b], Michalis Korakakis[b]

[a] Department of Informatics, University of Piraeus, Greece
[b] Department of Informatics, Ionian University, Greece

A R T I C L E I N F O

Article history:
Available online xxx

Keywords:
Location privacy
Online social networks
Private equality testing
Geosocial networks

A B S T R A C T

The wide adoption of smart phones has enabled Online Social Networks (OSNs) to exploit the location awareness capabilities offering users better interaction and context aware content. While these features are very attractive, the publication of users' location in an OSN exposes them to privacy hazards. Recently, various protocols have been proposed for private proximity testing, where users are able to check if their online friends are near, without disclosing their locations. However, the computation cost of the required cryptographic operations utilized in such protocols is not always efficient for mobile devices. In this paper we introduce a lightweight and secure proximity testing protocol, suitable for online mobile users. We show that our protocol is provably secure under the well-known factoring problem and we analyze its efficiency. Our results show that our approach outperforms other existing protocols, by significantly reducing the computational cost and making it practical for devices with limited resources. Finally, we demonstrate the applicability of our proposal in an actual OSN location-based, mobile application.

1. Introduction

Apart from being a modern trend in web applications, OSNs have totally reconstructed the way people interact and exchange information over the Internet [1]. OSNs allow mobile users to continuously interact with each other through mobile devices such as smart phones, tablets or smart watches. The amount and quality of exchanged information has made us rethink common information flow models. OSNs can be considered as critical information channels, since OSN users may exchange or publicly share a plethora of information. Based on the nature of the online application, this information may involve conversations with friends, user-oriented multimedia content (such as personal photos and videos), personal opinions, comments, habits, locations and user itinerary, to name some common examples. In addition, the users may also share personal information with the OSN providers, for example when registering to services. These novel information flows affect the role and the impact of individuals in such networks. In many cases the amount of personal and/or sensitive user-related information that is available to others, may enable other users or service providers to monetize this information and expose users to privacy-related threats. For example, an OSN may legitimately allow third parties to use such information, *e.g.* for targeted advertising, or allow them to mine anonymized graphs. Even if user data have been anonymized, it has been shown that re-identification algorithms may potentially de-anonymize user data and link them to particular users [2]. Unfortunately, many users share arbitrary information about their personal lives, greatly exposing themselves to privacy risks [3].

The gradual exploitation of location sensors of smart devices has turned modern OSNs into location-aware applications. In fact some of them are becoming so dependent on the geospatial information that can be regarded as a separate category, the so-called *geospacial social networks*. Further to just adding location tags to multimedia content, many of them try to bring their users closer by displaying their proximity to others in terms of distance. This trend is very common in dating focused OSNs such as MoMo, Plenty of Fish, WeChat, to name a few, where users not only see how many people are available, but their distance (or an approximation) to them.

Unfortunately, the location-awareness exposes OSN users to a number of risks [4–10]. In many cases these risks stem from poor implementations *e.g.* by displaying an estimation of the distance of a user from others. As it has been shown independently in [11,12] the provided obfuscation mechanisms can be trivially bypassed, exposing the exact location of the users. The exposure of a user's location can disclose a lot of sensitive information. An adversary could, for instance, deduce the victim's health condition (*e.g.* the victim is at a hospital/doctor), religious beliefs, (*e.g.* the victim is at a

* Corresponding author. Tel.: +30 2104142261.
E-mail addresses: pkotzani@unipi.gr (P. Kotzanikolaou), kpatsak@gmail.com, kpatsak@unipi.gr (C. Patsakis), emagos@ionio.gr (E. Magkos), p12kora@ionio.gr (M. Korakakis).

http://dx.doi.org/10.1016/j.comcom.2015.07.017
0140-3664/© 2015 Elsevier B.V. All rights reserved.

church/mosque), political beliefs (*e.g.* the victim is protesting at a demonstration), personal connections (*e.g.* two people are occasionally at the same place). Further to privacy exposure, the revelation of location can be used for cyber-bulling, cyber-stalking or even endanger their well-being. Unfortunately, users' awareness regarding the dangers of publicly sharing their location information through "check-ins" to online applications is low. For example the PleaseRobMe[1] application demonstrates the easiness to aggregate such information from various online sources and to deduce sensitive information about users' current location, or "recent empty homes".

Ideally, a *privacy-aware proximity testing service* in a *buddy finder* OSN application, receives the encrypted whereabouts from its subscribed users and notifies any two buddies (*e.g.*, Alice and Bob) when they are in proximity: Alice and Bob should learn a minimum amount of information, *e.g.*, whether they are in proximity, while the OSN provider or any other external entity should learn nothing. In *synchronous* proximity testing [13], which is very well-suited to OSN users equipped with mobile phones, Alice and Bob are concurrently online and run the steps of the proximity testing protocol in real-time.

To address the risks of location disclosure in buddy-finder services, with no trust assumptions, a line of recent works for synchronous private proximity testing, use a *decentralized* approach in the *public-key setting* [13–17], where a two-party protocol is executed between any two friends that wish to check their proximity. The outcome of such protocol can be as small as a single bit of information, *i.e.*, whether users are in the same vicinity. The most efficient approaches of the category employ, as their main building block, a two-party *Private Equality Testing (PET)* protocol between any two buddies, with private inputs representing geographic locations [13,15–17]

Our contribution: In this paper we are concerned with synchronous private proximity testing through *Private Equality Testing* (PET) for low min-entropy location data. We propose a very efficient PET protocol that provides unconditional location privacy for the initiator of the protocol, while the privacy for the responder is guaranteed under the intractability of the factoring problem. Our primitive is specifically suited for low-min entropy data, such as encrypted locations of users, and thus could be used as a building block for a proximity-testing application in a buddy-finder OSN service. Our protocol is very efficient since for a protocol run, it requires only one public-key exponentiation per user. We experimentally compare our protocol against other PET protocols with similar security properties, using Sage and the well-known cryptographic library MIRACL. Our results show that our protocol outperforms its peers in computation time, making it practical for typical off-the-self mobile devices.[2] Finally, we demonstrate the applicability of our proposal in an actual OSN location-based, mobile application.

Organization of this work: In Section 2 we review related work on private proximity testing. In Section 3 we present our protocol, which is a PET protocol based on the factoring problem, while in Section 4 we formally prove the security properties of our scheme. In Section 5 we give experimental results concerning the efficiency of the proposed scheme. In Section 6 we provide an overview of a mobile application we have developed to test the applicability of our protocol. Finally, Section 7 summarizes the contributions of the paper and discusses ideas for future work.

2. Related work

In the general literature for privacy preservation in location-based services (LBS), three general directions are taken to address the privacy problem [18–20]: (a) in approaches based on *location privacy*, the exact location of a user is obfuscated (*e.g.*, spatially cloaked or combined with noise); (b) solutions based on *identity privacy* protect the link between the user's location and her identity; (c) *hybrid* solutions, where, for example, locations are cloaked to include a region with a minimum number of users (*k*-anonymous queries). The above approaches mainly involve sporadic point-of-interest queries that are executed at an LBS provider, and thus are left out of scope. In addition, most of these privacy-aware approaches for LBS services assume that the users trust, either partially or fully, one or more system entities (*e.g.*, a trusted proxy, other system users, the LBS provider, a set of non-colluding third parties *etc.*). In addition, by considering solutions that incur minimal resource consumption for execution in a mobile cellphone, we leave out of scope solutions such as Private Information Retrieval (PIR) [21] and fully homomorphic encryption [22].

Private proximity testing in buddy-finder applications could be efficiently implemented, assuming that any two users share a symmetric cryptographic key [23–25] or a grid transformation key [11,26]. Besides the key management issues (*e.g.*, for establishing, storing and updating symmetric keys), the main problem with the symmetric-key setting is that friends typically trust each other for (some aspects of) their privacy. For this reason, in this paper we focus on public-key based solutions for private proximity testing, which do not assume the use of previously established symmetric keys.

Another privacy consideration for proximity testing is the *low-min entropy* of the location data; typically, the set of all possible locations cannot exceed 2^{40}. This precludes the use of deterministic public-key encryption (*e.g* [27]) to test equality/proximity, since this would allow a curious entity (buddy or third party) to exhaustively search the private input set. Furthermore, fully outsourcing the proximity/equality function to a proxy is succeptible to violation of location privacy. For example, the public-key encryption with equality testing (PKEET) primitive [28,29] allows a proxy to execute on behalf of two users A and B, a function $Test(c_A, c_B)$ over two ciphertexts c_A, c_B, that are probabilistically encrypted with different public keys pk_A, pk_B, in order to check whether they contain the same message. This is not a suitable solution for testing equality on encrypted locations, due to the low min-entropy domain; a malicious proxy having access to the *Test* function and the users' public keys, is able to exhaustively select candidate messages, encrypt them with a user's public key and then use the PKEET test function to perform offline message (location) recovery.

Efficient, synchronous privacy-preserving proximity testing in buddy-finder applications, with no trust assumptions, can be typically done in either one of two ways: First, a user's location is approximated with one or more grid cells of sufficient size, where each cell is 1–1 mapped to a unique index number; then, proximity is decided using a Private Equality Testing (PET) protocol to test equality of the private indexes (*e.g.*, [13,15,17]). Note that, at a high level, a two-party PET protocol constitutes an efficient instantiation of a multi-party protocol for *private set intersection* (PSI) (*e.g.*, [30–33]). Alternatively, a proximity testing protocol can calculate the distance of exact user locations (*e.g.*, [13,14]). Solutions of the first category have been considered as more efficient [16,34]. Furthermore, it was recently shown that user privacy can be violated in any scheme that reveals approximate distance information to the service provider [35].

The Nearby Friend protocol of Chatterjee et al.: Chatterjee et al. [15] proposed the Nearby Friend (NFP) protocol, where Alice (the initiator of a protocol run) can efficiently determine whether her friend, Bob (the responder), is at a nearby location or not, while Bob learns nothing. The set of locations is mapped to a multiplicative group of points on an elliptic curve of prime order, defined over a finite field of prime order. In the NFP protocol, the users do not disclose their location information to each other (in case of inequality) while the service provider or any other entity learns nothing about the users' locations. The protocol is based on the Diffie–Hellman type of the simple password exponential key exchange (SPEKE) protocol [36]. Alice's

[1] http://pleaserobme.com.

[2] We consider devices capable to run smartphone versions of typical OSN applications; we do not require the processing capabilities of high-end, high-cost devices.

location privacy is unconditional; even if an adversary is unbounded, the location of Alice cannot be disclosed. The location of Bob in the NFP scheme is based on the hardness of the Decisional DH (DDH) assumption. The computational cost for each equality test is two exponentiations per user.

The EG-PET protocol of Narayanan et al.: Narayanan et al. [13] propose an equality testing scheme, suitable for private proximity testing in LBS applications. Alice uses an ElGamal key to encrypt her location and send it to Bob through an authenticated channel. Bob then “hides” his location to Alice’s encryption and replies to Alice, who is then able to decrypt with her public key and verify if the two locations are equal or not (but nothing more). The EG-PET protocol is *asymmetric* since Alice first learns the outcome of the equality testing and may then let Bob to learn the result (in [15] only Alice learns the outcome of the equality test). Alice’s privacy is based on the DDH assumption, while Bob’s is unconditional. The computational cost is three exponentiations for Alice and four for Bob.

The VPET protocol of Saldamli et al. [34] builds upon the protocol of Narayanan et al. to propose a PET protocol which decreases the use of cryptographic primitives by blinding the values through a simple geometric representation of the values. In addition, Lin et al. [16] build on the protocol of [13] by capturing location tags (*i.e.*, ephemeral keys corresponding to a given time/location) on GSM cellular networks, while Zheng et al. [37] use Bloom filters to represent the location tags efficiently.

The DH-PET protocol of Magkos et al.: Magkos et al. [17] propose a PET protocol, based on Diffie–Hellman DH key agreement [38], to allow two peers, say Alice and Bob, to securely test the equality of their private, low-entropy input data g_A and g_B, respectively, without the involvement of any third party. The DH-PET protocol provides unconditional input privacy against external observers for both parties. It also provides privacy of the users against each other based on the hardness of Discrete Logarithm Problem (DLP). The protocol requires two exponentiations per user.

A comparison of the existing PET protocols (including the proposed protocol) in terms of security and efficiency properties, is given in Section 7.

3. A PET protocol based on factoring (FP-PET)

Similar to the PET protocols presented above [13,15,17], our protocol is also suitable for low-min entropy data, since it is resistant to exhaustive offline message recovery attacks. In addition, as it is going to be shown in Section 5, our protocol is far more efficient than the others since it requires only one modular exponentiation for each user. The security of our PET protocol, is based on the intractability of the factoring problem (FP).

3.1. Threat model

We assume that users of the protocol adhere to a *honest but curious* model (HBC) (also known as semi-honest), in that they abide to the rules of the protocol while trying to learn as much as possible about the private data of the other users. We assume probabilistic polynomial time (PPT) passive adversaries that are polynomially bounded and do not have the ability to break the underlying cryptographic primitives used (*i.e.* reverse hash functions or break the factoring problem). We also assume that an adversary is able to monitor all the traffic exchanged within the protocol. We do not consider active attacks; we assume that the messages exchanged in a protocol run are authenticated and integrity protected, thus the adversary is not able to modify or inject fake messages pretending to originate from another legitimate user.

3.2. Setup

Let $L = \{\ell_1, \ell_2, \dots, \ell_k\}$ be a discrete, finite, low-min entropy input set (*e.g.* $|L| < 2^{40}$), containing data representing all possible locations (*e.g.*, GPS coordinates). The service provider (SP) will compute a set $\mathcal{L} = \{l_1, l_2, \dots, l_k\}$, where each element l_i is a unique odd number of small length (*e.g.* $|l_i| < 40$ bit). Then, the SP will use an “1–1” random mapping $f : L \rightarrow \mathcal{L}$ to uniquely assign each location (element of L) to a unique element in the set $\mathcal{L}$.

Note that both sets L and $\mathcal{L}$ need to be constructed only once by the SP and can be valid throughout the lifetime of the system. The SP will publish the sets L, $\mathcal{L}$, and their mapping f to make the assignment mechanism publicly verifiable. Note that the assignment mechanism will be transparent to all users. For the sake of simplicity, instead of saying that a user chooses an input $\ell_i \in L$, we will directly say that the user chooses $l_i \in \mathcal{L}$. Furthermore, let $H(\cdot)$ denote a secure cryptographic hash function of sufficient length, such as SHA256. Let “|” denote concatenation of two messages.

Each user U selects two primes p_u, q_u of sufficient length and computes and publishes $n_u = p_u \cdot q_u$. The prime numbers p_u, q_u (the factoring of n_u) are kept secret. To boost the efficiency and practicality of the scheme we propose the use of safe primes, that is $p_u = 2p' + 1$ and $q_u = 2q' + 1$. This recommendation makes $\phi(n_u) = 4p'q'$ with p', q' being primes. Therefore, all small odd numbers are coprime with $\phi(n_u)$. This allows us to use a function $f(\cdot)$ which does not need to be a “1–1” mapping; a cryptographic hash function of suitable length can be used to perform the mapping more efficiently without needing to cache or precompute the mapping of locations. For instance users could use the mapping $l_i = 2f(\ell_i) + 1$, where f is a cryptographically secure function. Assume that, for a given time period, a user U is located within $\ell_u \in L$, represented as l_u in the set $\mathcal{L}$. Then U will compute an “RSA-like” key pair as follows: set $d_u \equiv l_u$ and then compute $e_u : d_u \cdot e_u = 1 \mod \varphi(n_u)$. Since each element $l_i \in \mathcal{L}$ is a prime number, then it is easy to see that every user will be able to compute, modulo $\varphi(n_u)$, for any element l_u, the inverse element e_u and thus efficiently compute the key pair (d_u, e_u). Both keys (d_u, e_u) are kept private. This key pair will be used in the PET protocol, as long as U is within the same location $\ell_u \in L$. When the user moves to another location ℓ'_u, she will compute new keys (d'_u, e'_u) in the same way.

3.3. The FP-PET protocol

Alice (the initiator of the protocol) has published $n_A = p_A \cdot q_A$ (where p_A, q_A are kept private) and she currently uses the private key pair $d_A (\equiv l_A)$ and e_A, computed as described in Section 3.2. In the same way, Bob has published $n_B = p_B \cdot q_B$ and he currently uses the private key pair $d_B (\equiv l_B)$ and e_B.

Input: Alice has private input $l_A \in \mathcal{L}$ and Bob has private input $l_B \in \mathcal{L}$.

Output: Alice learns whether $l_A = l_B$ and Bob learns nothing (asymmetry). If Alice chooses to reveal this to Bob, Bob will also learn whether $l_B = l_A$.

The steps of the protocol are the following:

Step 1: Alice picks a random integer r_A of high entropy (say, 1024 bit), computes: $c_A = r_A^{l_A} \mod n_B$ and sends it to Bob.

Step 2: Bob computes $x = c_A^{e_B} \mod n_B$ and $c_B = H(x)$ and sends c_B to Alice.

Step 3 (equality test for Alice): Alice checks whether $c_B \overset{?}{=} H(r_A \mod n_B)$. If the equality holds, Alice is convinced that $l_A = l_B$. If not, Alice learns nothing about Bob’s private input.

Step 4 (equality test for Bob – optional): If the verification of Step 3 was successful, Alice computes $y = H((r_A|l_A) \mod n_B)$ and sends it to Bob. Otherwise, Alice sends Bob a random nonce.

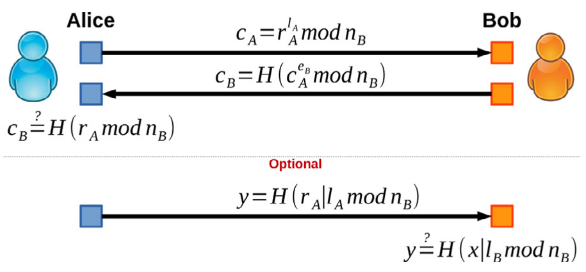
Alice
Bob
$c_A = r_A^{l_A} mod\, n_B$
$c_B = H(c_A^{e_B} mod\, n_B)$
$c_B \overset{?}{=} H(r_A mod\, n_B)$
Optional
$y = H(r_A | l_A mod\, n_B)$
$y \overset{?}{=} H(x | l_B mod\, n_B)$

Fig. 1. The proposed protocol.

Bob will check whether $y \overset{?}{=} H((x|l_B) \bmod n_B)$ and learn whether $l_B = l_A$. If the equality check fails, then Bob learns nothing about Alice's location.

The protocol is illustrated in Fig. 1.

3.4. Protocol correctness

Let us assume that $l_A = l_B$. Then, in Step 2, Bob computes $x = c_A^{e_B} = ({r_A}^{l_A})^{e_B} = r_A^{l_B e_B} = r_A \bmod n_B$. Thus, in case of equality, the verification of Alice in Step 3 will always be successful. In addition, modulo n_B, $r_A|l_A$ will be equal to $x|l_B$, in case of equality, and so the verification from Bob in Step 4 will also be successful.

4. Security analysis

We consider both external and internal adversaries. An external adversary $\mathcal{A}^{ext}$ represents all entities other than the users running the protocol, including external attackers and the SP. The goal of an external adversary is, by using as input the exchanged messages of a protocol run, to: (i) learn information about the private input of any of the parties that participate in the protocol and/or (ii) decide whether the private input of two users running the protocol are equal or not. Internal adversaries represent an honest but curious user running the protocol. The goal of an internal adversary $\mathcal{A}^{int}$, is to reveal the private information of the other party, in case of inequality.

Definition 4.1. A function $\nu(\cdot)$ is negligible in x, or just negligible, if for every positive polynomial $p(\cdot)$ and any sufficiently large x it holds that:

$$\nu(x) \leq \frac{1}{p(x)}$$

4.1. Security against external adversaries

We examine the capabilities of an external adversary attempting to either learn the private input of the users or to learn the outcome of the protocol run. For simplicity, let Π denote the PET protocol of Section 3.3.

4.1.1. Private input indistinguishability against external adversaries

We formalize private input indistinguishability against external eavesdroppers by a security experiment $DistExp^{ext}$ in which $\mathcal{A}^{ext}$ has access to an oracle $\mathcal{O}^{dist}$ that on input: the low-entropy set of all possible private input $\mathcal{L}$, the public parameters of two non-compromised users Alice and Bob n_A, n_B and a protocol run $[c_A, c_B, y]$, is attempting to learn information about the private input of Alice and/or Bob. If $\mathcal{O}^{dist}$ is able to distinguish the private input of either party (l_A and/or l_B) from the set $\mathcal{L}$ using the given input (where $|\mathcal{L}|$ is the security parameter), then the output of the experiment is 1, else the output is 0.

Definition 4.2. [Private input indistinguishability against external adversaries] Π provides private input indistinguishability against an external adversary if $\forall$ adversary $\mathcal{A}^{ext}$, $\exists$ a negligible function ν such that:

$$Advantage(\mathcal{A}^{ext}) = \left| Pr[DistExp^{ext}(|\mathcal{L}|) = 1] - \frac{1}{|\mathcal{L}|} \right| = \nu(|\mathcal{L}|)$$

Theorem 1. *Π provides unconditional private input indistinguishability for Alice against any external adversary.*

Proof. Consider an external adversary who is *not* polynomially bounded, and is able to break the factoring problem for n_B, thus revealing p_B, q_B. Then $\mathcal{A}^{ext}$ will be able to compute for every element $l_i \in \mathcal{L}$ its inverse element $e_i \bmod \phi(n_B)$. Now the question for $\mathcal{A}^{ext}$ is to distinguish which key pair (l_i, e_i) corresponds to the key pair $(d_A(\equiv l_A), e_A)$ of Alice. Since $\forall l_i \in \mathcal{L}$, $\exists r_i : c_A = r_i^{l_i} \bmod n_B$, then the equation can be solved for every l_i, with each solution being equally probable with the others. Hence, the private input c_A of Alice may contain any element of $\mathcal{L}$ with equal probability and $\mathcal{A}^{ext}$ has negligible advantage to distinguish the correct one. □

Theorem 2. *Π provides private input indistinguishability for Bob against a PPT external adversary, if the factoring problem is hard and $H(\cdot)$ is a cryptographically secure hash function.*

Proof. Bob uses his private key e_B, (the inverse element of the low-min entropy private input l_B) and computes $x = c_A^{e_B} = r_A^{l_A e_B} \bmod n_B$. Then, Bob sends the hash value $c_B = H(x)$. The adversary cannot recover x from c_B since H is assumed to be a secure hash function. Therefore, the problem for $\mathcal{A}^{ext}$ is to distinguish which element $l_i \in \mathcal{L}$ is Bob's private input. To distinguish this, $\mathcal{A}^{ext}$ must be able to find the inverse of every element $l_i \in \mathcal{L}$, to get $e_i \bmod \phi(n_B)$ and then find which is the correct one, by using c_B to verify the test. If $H(c_A^{e_i} \bmod n_B) = c_B$, then $\mathcal{A}^{ext}$ decides that the test key pair (l_i, e_i) corresponds to (l_B, e_B). However, since $\mathcal{A}^{ext}$ is polynomially bounded and n_B is of sufficient length, it is not possible for $\mathcal{A}^{ext}$ to factor n_B and consequently find the inverse element $e_i \bmod \phi(n_B)$ of any element $l_i \in \mathcal{L}$. Thus, it is impossible for an external adversary to learn the private input of Bob by brute-forcing all possible input elements, under the intractability of the factoring assumption. □

4.1.2. Private input equality undecidability against external adversaries

We formalize private input equality undecidability against external eavesdroppers by a security experiment $EqualExp^{ext}$ in which $\mathcal{A}^{ext}$ has access to an oracle $\mathcal{O}^{equal}$ that on input: the low-entropy set of all possible private input $\mathcal{L}$, the public parameters of Alice and Bob n_A, n_B and a protocol run $[c_A, c_B, y]$, is attempting to learn whether the equality test of Alice and Bob was successful or not. If $\mathcal{O}^{equal}$ is able to decide that the PET between the two users was successful (with security parameter $|\mathcal{L}|$), then the output of the experiment is 1, else the output is 0.

Definition 4.3 (Private input equality undecidability against external adversaries)**.** Π provides private input equality undecidability against an external adversary if $\forall$ PPT adversary $\mathcal{A}^{ext}$, $\exists$ a negligible function ν such that:

$$Advantage(\mathcal{A}^{ext}) = \left| Pr[EqualExp^{ext}(|\mathcal{L}|) = 1] - \frac{1}{2} \right| = \nu(|\mathcal{L}|)$$

Theorem 3. *Π provides private input undecidability against an external adversary if $H(\cdot)$ is a cryptographically secure hash function.*

Proof. From Theorem 1 it is clear that $\mathcal{A}^{ext}$ learns nothing from c_A. Thus, the adversary must use c_B and y to decide whether the equality test was successful. Recall that in case of equality in Step 3 of the protocol, Alice will send to Bob $y = H((r_A|l_A) \bmod n_B)$, else y is a random nonce. Since $H(\cdot)$ is assumed to be a cryptographically secure hash function, $\mathcal{A}^{ext}$ cannot invert y and c_B, or decide if y is random or not. □

4.2. Security against internal adversaries

We examine the capabilities of an internal curious adversary, attempting to learn the private input of the other party in case of input test inequality.

4.2.1. Private input indistinguishability against internal adversaries

The security experiment is similar to the one described in Section 4.1 for external adversaries with the following difference. In the case of internal adversaries, the experiment formalizes one curious participant of a protocol run (either a curious initiator – Alice, or a curious responder – Bob), attempting to learn the private input of the other party (respectively of Bob or Alice). Thus, the security experiment $DistExp^{int}$ will take, in addition to the input given to the security experiment $DistExp^{ext}$, the private keying material of the curious user; either Alice's ($\mathcal{K} = \{p_A, q_A, l_A, e_A\}$) or Bob's ($\mathcal{K} = \{p_B, q_B, l_B, e_B\}$). If $\mathcal{O}^{dist}$ is able to distinguish the private input of the other party from the set $\mathcal{L}$ (either of Bob or of Alice respectively) then the output of the experiment is 1, else the output is 0.

Definition 4.4 (Private input indistinguishability against internal adversaries)**.** Π provides private input indistinguishability against an internal honest-but-curious adversary if $\forall$ PPT adversary $\mathcal{A}^{int}$, $\exists$ a negligible function ν such that:

$$Advantage(\mathcal{A}^{int}) = \left| Pr[DistExp^{ext}(|\mathcal{L}|) = 1] - \frac{1}{|\mathcal{L}|} \right| = \nu(|\mathcal{L}|)$$

Theorem 4. *Π provides to Alice unconditional private input indistinguishability against Bob.*

Proof. Since now Bob is assumed to be compromised by the adversary, $\mathcal{K} = \{p_B, q_B, l_B, e_B\}$ and the experiment will output 1 if $\mathcal{O}^{dist}$ succeeds to distinguish l_A from the set $\mathcal{L}$.

The proof is analogous to the proof of Theorem 1. Since in the proof of Theorem 1 we examined a polynomially unbounded adversary, the knowledge of the factoring of n_B does not give an additional advantage to Bob. Again, even if $\mathcal{A}^{int}$ is able to compute the inverse element e_i mod $\phi(n_B)$ for every element $l_i \in \mathcal{L}$ it is still not possible to distinguish which key pair corresponds to l_A, e_A of Alice, since $\forall\, l_i \in \mathcal{L}$, $\exists\, r_i : c_A = r_i^{l_i}$ mod n_B. Recall that in case the equality test of Alice on Step 3 of the protocol fails, then Alice sends as y a random nonce to Bob, thus Bob gains no additional information about l_A. □

Theorem 5. *Π provides private input indistinguishability to Bob against Alice, if the factoring problem is hard and $H(\cdot)$ is a cryptographically secure hash function.*

Proof. Since in this case Alice is assumed to be compromised by the adversary, $\mathcal{K} = \{p_A, q_A, l_A, e_A\}$ and the experiment will output 1 if $\mathcal{O}^{dist}$ succeeds to distinguish l_B from the set $\mathcal{L}$. The proof is essentially the same as in Theorem 2, since $\mathcal{K}$ does not provide any additional information for the prime factors p_B, q_B of Bob to Alice, in comparison with an external adversary. □

5. Efficiency analysis

To verify the efficiency of our protocol, we have implemented our protocol, the protocol of Magkos et al. [17] and the protocol of Narayanan et al. [13][3] using two different cryptographic approaches. The first implementation uses the Sage[4] 6.3 mathematics software system which is based on Python. In this case, all the protocols have been implemented over integers, as in the original papers. Since the protocols of Magkos et al. and of Narayanan et al. are based on the discrete log problem, their implementation over elliptic curves would be an obvious optimization leading to better computation and communication times. However, Sage implementations over elliptic curves are not optimized to make reliable comparisons (actually, the elliptic curve version of each protocol using Sage, exhibited lower performance that its integer version). Therefore, we have also implemented the protocols in C++ using the MIRACL[5] crypto library, which provides very optimized implementations of cryptographic primitives and operations over elliptic curves. In both test cases, our protocol has been implemented over integers. While our protocol could also be implemented over elliptic curves in MIRACL, its factoring-based nature would lead to elliptic curves over large complex numbers, rendering the features of elliptic curves useless.

Table 1
Communication cost for 1024 bits of public key security.

	Alice	Bob
Narayanan et al.	1024	2048
Magkos et al.	1024	1280
Proposed protocol	1024	256

All the experiments were made on a 64 bit Linux machine, with 3.13.0-36 kernel running on an Intel Core i3-2100 at 3.10 GHz with 4 GB of RAM. Our test implementations of all the examined protocols can be found at Github[6]. For each protocol, we run 1000 tests and the mean execution time was then computed for each player of the protocol. The reported measurements do not consider any communication or other kind of delay and show the absolute computation time required for the cryptographic operations utilized in each protocol without multithreading.

In the case of the Sage based implementations, we measured for each protocol the required computation time for two scenarios; using 1024 bit keys and 2048 bit keys respectively, for each examined protocol. In the case of the MIRACL based implementations, our protocol is still measured for 1024 and 2048 bit settings, while the elliptic curve versions of the discrete log based protocols are tested for 160 and 224 bit settings. These settings are commonly believed[7] to provide the same security level with typical RSA on 1024 and 2048 bits respectively.

The computational cost of our experimental results is illustrated in Figs. 2 and 3, while Table 1 illustrates the communication cost. Clearly, our protocol outperforms its peers in terms of execution time, even when compared with their elliptic curve versions. This is expected since our PET requires only 2 exponentiations for a protocol run (one for each user), while the protocol of [17] (as well as the protocol of [15]) require 4 exponentiations and the protocol of [13] 7 exponentiations in total (see Table 3 for a complete comparison of the protocol characteristics). However, our protocol exhibits even lower computation time than the expected. The efficiency of our protocol stems from the fact that Alice has to perform a small exponentiation; locations can safely be represented as small odd integers, while Bob has to perform a single exponentiation. Although the exponent e_B might be large, the full exponentiation can be significantly improved using the Chinese remainder theorem and the known factorization of n_B. Regardless of whether the underlying algorithms are running in Python or C++, and the features of each framework, the proposed algorithm manages to outperform the other protocols in all settings and timings of each individual entity. Thus our protocol can be efficiently used in mid-range mobile devices without the mobile users experiencing intolerable delays.

[3] The computation efficiency of the NFP protocol [15] is similar to the Magkos et al. protocol, since they require the same number of exponentiations.
[4] http://www.sagemath.org/.
[5] http://www.certivox.com/miracl/.
[6] https://github.com/kpatsakis/Factoring_based_PET.
[7] https://www.nsa.gov/business/programs/elliptic_curve.shtml.

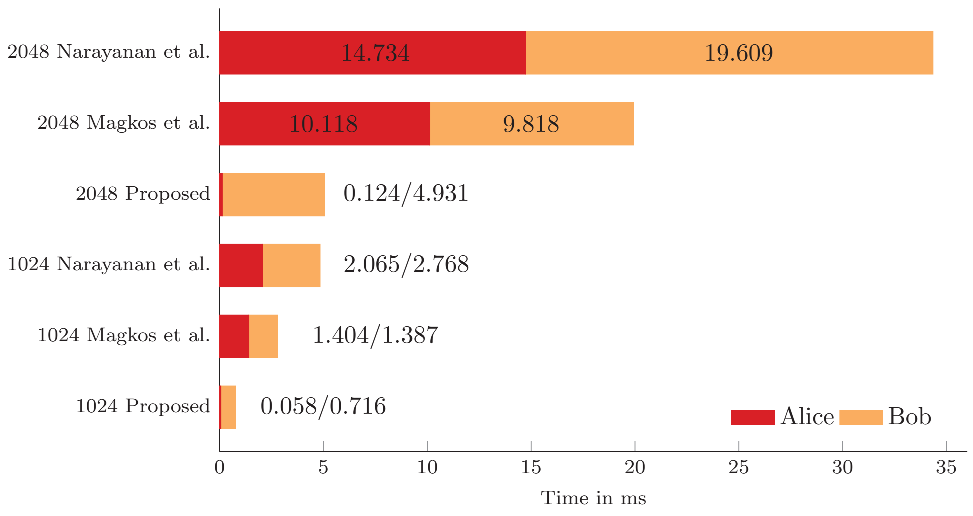

Fig. 2. Summary of the computational cost (time in ms) for the examined PET protocols for 1024 and 2048 modulo settings using the Sage library.

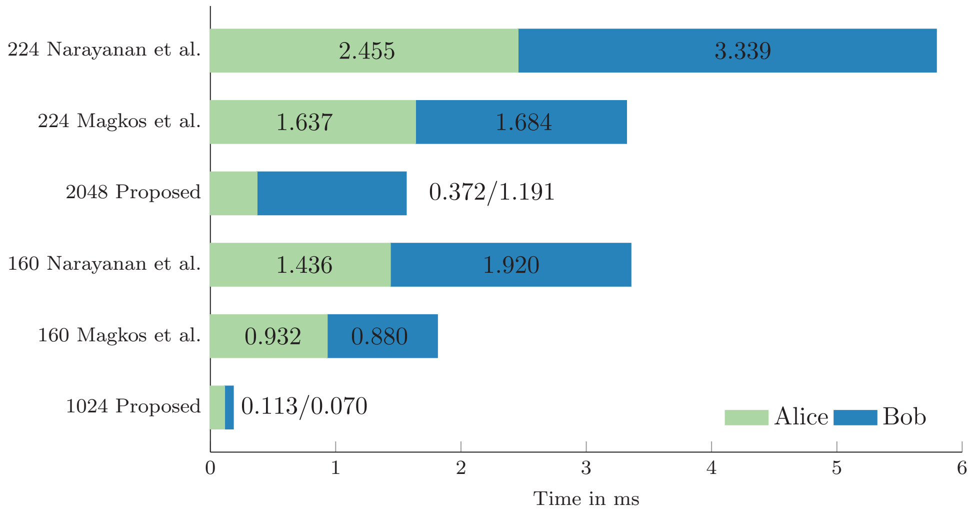

Fig. 3. Comparison of the computation time (in ms) using the MIRACL library. The discrete log based protocols (Magkos et al. and Narayanan et al.) are implemented over elliptic curves; our protocol is implemented over integers with key size of similar security level.

6. Prototype development

To illustrate the usability of our protocol in a real-world environment, we have developed an Android application. Our application is a typical buddy-finder location-based service for Facebook.

Functionality: When the user logs in for the first time, the client registers itself with the Google Cloud Messaging Platform. The result of this process is the creation of a registration number that enables the app to successfully forward and receive messages. The underlying OSN in our case is Facebook from which we import profile information of the user and his friends to perform proximity testing. On the initialization, the app runs our protocol with all the available Facebook friends. The friends are then categorized into those who are in *proximity*: friends that are using the app, they are online and they are in proximity according to our protocol; *online*: friends who are online but are not close, and finally *offline*: friends who are not currently using the app. From the application settings, the user can determine the frequency (epoch), with which the app executes the protocol. Additionally, may decide whether she wants to execute Step 4 of the protocol, *i.e.*, acknowledge her proximity to her friend(s).

Table 2
Battery consumption.

	500 friends (J)	1000 friends (J)
Alice (Step 1)	4.3	9.2
Bob (Step 2)	8.4	22.0
Alice (Step 3)	0.4	1.2

Cryptography: For the implementation of the FP-PET protocol we use Spongy Castle[8], an as-is version of Bouncy Castle[9] for Android.

Client-Server: The communication module is based on the Google Cloud Messaging platform. The selection of this service was made on one of its basic features, it enable developers to create applications that send and receive data without having to worry about implementing querying and message delivering operations. Furthermore, we provide the cloud infrastructure with a third party web server that acts as a mediator, by storing and distributing various critical

[8] https://rtyley.github.io/spongycastle/.
[9] https://www.bouncycastle.org/.

Table 3
Comparison of our scheme with the schemes of [13–15,17].

Protocol	Aim	Relies on	Equality testing	Security		Efficiency
				vs insiders	vs outsiders	
Pierre [14]	Initiator-only: Alice checks proximity	CGS [40]	Bob sends to Alice the (tentative) DH key	Alice:Uncond. Bob: DDH	Alice:Uncond. Bob: DDH	Alice: 6 exp+3 DL Bob: 6 exp
NFP [15]	Initiator-only: Alice checks proximity	DH-based SPEKE [36]	Bob sends to Alice the (tentative) DH key	Alice:Uncond. Bob: DDH	Alice:Uncond. Bob: DDH	2 exp/user
EG-PET [13]	Asymmetric: Alice may let Bob learn proximity result	ElGamal encryption	Decryption of a properly constructed ciphertext	Alice: DDH Bob: Uncond.	Both:Uncond.	Alice: 3 exp Bob: 4 exp
DH-PET [17]	Asymmetric: Alice may let Bob learn proximity result	DH-based SPEKE [36]	Implicit, with symmetric challenge-response	Both: DLP	Both:Uncond.	2 exp/user
FP-PET	Asymmetric: Alice may let Bob learn proximity result	RSA-based	Decryption of an RSA-based challenge	Alice:Uncond. Bob:FP	Alice:Uncond. Bob: FP	1 exp/user

information along with the forwarded message, between the device and the Google Cloud. The push notification mechanism, which is implemented by the Google Cloud platform, can significantly cut down any data usage and work load costs for both the app and the server by removing the need for constant data pulling effectively, leading to, *e.g.*, better battery life and less network bandwidth consumption.

Grid: For the sake of simplicity, the grid creation is based on the implementation used in [13], *e.g.* a 3-way overlapping grid system. This approach effectively solves any limitations that are associated with spherical coordination mapping.

Battery consumption: In order to test the efficacy of our protocol we use PowerTutor [39], one of the most accurate energy profiler application for mobile phones, to measure its power consumption. The results about the energy consumption on a Nexus 10 device, using RSA with key size of 1024 bits are illustrated in Table 2. Totally, if Alice has 500 friends[10], then to test to whom she is close, she would consume 4.7 J while Bob would consume 8.4 J to respond to 500 location queries. Similarly, for the case of 1000 friends, Alice and Bob would consume 10.4 and 22 J respectively. A common off-the-self smartphone would have a battery of 5.45 Wh which accounts for 19,620 J. Therefore, Alice could make 4174 protocol executions per charge for 500 friends or 1886 executions for 1000 friends. Similarly, Bob could make 2335 and 891 executions respectively. Clearly, the above indicate that users could check their proximity to others regularly, without the risk of draining their battery. Clearly, it is high improbable that a user would have all his friends on line to perform so many tests simultaneously, so the actual consumption would be far less.

7. Discussion and comparison with related work

For a holistic evaluation of the proposed protocol against similar approaches, we summarize their similarities and differences regarding the aim, the security and the efficiency characteristics. Table 3 provides such a comparison with the protocols of [13–15,17], since all these protocols are suitable for low entropy (location) input data.

While all these protocols aim at proximity testing, the original version of the NFP protocol [15] allows only the initiator (Alice) of the protocol to learn the proximity result, while the responder (Bob) learns nothing. All other protocols are asymmetric in the sense that Alice first learns the result and optionally can run an additional round to also let Bob learn the outcome of the proximity test. The NFP protocol could also be extended to provide asymmetry but in its current version it requires a second protocol run with Bob as the initiator.

Concerning the security properties of the examined protocols, the schemes of [13,15,17] are based on a discrete log setting with variations on their security assumptions. The EG-PET protocol of [13] is based on ElGamal encryption and its security relies on the Decisional Diffie–Hellman (DDH) problem. The DH-PET protocol of [17], and the NFP of [15] are based on variations of DH key establishment. However in [15] Alice actually sends the (tentative) DH key to Bob, while in the DH-PET of [17] the users do not exchange any key but they implicitly confirm if their keys (and consequently their private locations) are equal. For this reason the security of the DH-PET protocol [17] relies on the Discrete Log Problem (DLP), while the NFP protocol [15] relies on the weaker DDH problem. The Pierre protocol [14] is based on the CGS encryption protocol [40] so the decryption relies on the computation of discrete log. Our protocol is is the first PET protocol that is based on the factoring problem (FP). In all protocols the location privacy of one of the parties (Alice or Bob) is unconditional.

In terms of computational efficiency, our protocol is the fastest, since it only requires one exponentiation per user (in total two), in comparison with NFP and DH-PET requiring 4 exponentiations and EG-PET requiring 7 exponentiations at total. Although discrete log based schemes are expected to be more efficient when implemented over elliptic curves in comparison with factoring based schemes which do not have any advantage on elliptic curves, our results show that our protocol performs much faster that its peers. Our protocol can securely use small exponents as private locations and this leads to very fast computation times, as shown in Section 5. Despite the library and setting used, our protocol performs much faster than its peers, from 2.2 up to 6.8 times less computation time. Thus our protocol can be practically implemented even in low or mid range mobile devices for mobile OSN users. We plan to make our test application available for a real OSN environment to actually demonstrate the feasibility of our proximity protocol. We also plan to extend our protocol to also support off-line proximity testing without sacrificing its privacy properties.

Acknowledgments

This work was supported by the European Commission under the Horizon 2020 Programme (H2020), as part of the *OPERANDO* project (Grant Agreement no. 653704) and is based upon work from COST Action *CRYPTACUS*, supported by COST (European Cooperation in Science and Technology).

References

[1] N.K. Baym, Personal Connections in the Digital Age, Polity, 2010.
[2] A. Narayanan, V. Shmatikov, De-anonymizing Social Networks, in: Proceedings of 30th IEEE Symposium on Security and Privacy, IEEE, 2009, pp. 173–187.

[10] Note that the average Facebook user has far less friends http://www.pewresearch.org/files/2014/01/Survey-Questions_Facebook.eps.

[3] B. Debatin, J.P. Lovejoy, A.-K. Horn, B.N. Hughes, Facebook and online privacy: attitudes, behaviors, and unintended consequences, J. Computer-Med. Commun. 15 (1) (2009) 83–108.
[4] C. Ruiz Vicente, D. Freni, C. Bettini, C.S. Jensen, Location-related privacy in geo-social networks, IEEE Internet Comput. 15 (3) (2011) 20–27.
[5] W. He, X. Liu, M. Ren, Location cheating: a security challenge to location-based social network services, in: Proceedings of 31st International Conference on Distributed Computing Systems (ICDCS), IEEE, 2011, pp. 740–749.
[6] V. Kostakos, J. Venkatanathan, B. Reynolds, N. Sadeh, E. Toch, S.A. Shaikh, S. Jones, Who's your best friend?: targeted privacy attacks in location-sharing social networks, in: Proceedings of the 13th International Conference on Ubiquitous Computing, UbiComp'11, 2011, pp. 177–186.
[7] T. Pontes, M.A. Vasconcelos, J.M. Almeida, P. Kumaraguru, V. Almeida, We know where you live: privacy characterization of foursquare behavior, in: Proceedings of International Conference on Ubiquitous Computing, UbiComp, 2012, pp. 898–905.
[8] B. Carbunar, M. Rahman, N. Pissinou, A. Vasilakos, A survey of privacy vulnerabilities and defenses in geosocial networks, IEEE Commun. Mag. 51 (11) (2013) 114–119.
[9] C. Patsakis, A. Zigomitros, A. Papageorgiou, A. Solanas, Privacy and security for multimedia content shared on osns: issues and countermeasures, Comput. J. (2014) bxu066.
[10] C. Patsakis, A. Zigomitros, A. Solanas, Analysis of privacy and security exposure in mobile dating applications, in: Proceedings of the International Conference on Mobile, Secure and Programmable Networking (MSPN'2015), Springer, 2015. (to appear).
[11] M. Li, H. Zhu, Z. Gao, S. Chen, K. Ren, L. Yu, S. Hu, All your location are belong to us: breaking mobile social networks for automated user location tracking, arXiv preprint arXiv:1310.2547 (2013).
[12] G. Qin, C. Patsakis, M. Bouroche, Playing hide and seek with mobile dating applications, in: N. Cuppens-Boulahia, F. Cuppens, S. Jajodia, A. Abou El Kalam, T. Sans (Eds.), ICT Systems Security and Privacy Protection, Volume 428 of IFIP Advances in Information and Communication Technology, Springer, Berlin, Heidelberg, 2014, pp. 185–196.
[13] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, D. Boneh, Location privacy via private proximity testing, in: NDSS, The Internet Society, 2011.
[14] G. Zhong, I. Goldberg, U. Hengartner, Louis, Lester and Pierre: three protocols for location privacy, in: Privacy Enhancing Technologies, Springer, 2007, pp. 62–76.
[15] S. Chatterjee, K. Karabina, A. Menezes, A new protocol for the nearby friend problem, in: Proceedings of the 12th IMA International Conference on Cryptography and Coding, Springer-Verlag, 2009, pp. 236–251.
[16] Z. Lin, D.F. Kune, N. Hopper, Efficient private proximity testing with GSM location sketches, in: Financial Cryptography and Data Security, Springer, 2012, pp. 73–88.
[17] E. Magkos, P. Kotzanikolaou, M. Magioladitis, S. Sioutas, V.S. Verykios, Towards secure and practical location privacy through private equality testing, in: Privacy in Statistical Databases, Springer, 2014, pp. 312–325.
[18] C.A. Ardagna, M. Cremonini, S.D.C. di Vimercati, P. Samarati, Privacy-enhanced location-based access control, in: Handbook of Database Security, Springer, 2008, pp. 531–552.
[19] E. Magkos, Cryptographic approaches for privacy preservation in location-based services: a survey, Int. J. Inf. Technol. Syst. Approach 4 (2) (2011) 48–69.
[20] M. Wernke, P. Skvortsov, F. Dürr, K. Rothermel, A classification of location privacy attacks and approaches, Pers. Ubiquitous Comput. 18 (1) (2014) 163–175.
[21] B. Chor, E. Kushilevitz, O. Goldreich, M. Sudan, Private information retrieval, J. ACM 45 (6) (1998) 965–981.
[22] C. Gentry, A Fully Homomorphic Encryption Scheme (Ph.D. thesis), Stanford University, 2009.
[23] L. Šikšnys, J.R. Thomsen, S. Šaltenis, M.L. Yiu, O. Andersen, A location privacy aware friend locator, in: Advances in Spatial and Temporal Databases, Springer, 2009, pp. 405–410.
[24] L. Siksnys, J.R. Thomsen, S. Saltenis, M.L. Yiu, Private and flexible proximity detection in mobile social networks, in: Proceedings of Eleventh International Conference on Mobile Data Management (MDM), IEEE, 2010, pp. 75–84.
[25] S. Mascetti, D. Freni, C. Bettini, X.S. Wang, S. Jajodia, Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies, VLDB J. 20 (4) (2011) 541–566.
[26] K.P. Puttaswamy, S. Wang, T. Steinbauer, D. Agrawal, A. El Abbadi, C. Kruegel, B.Y. Zhao, Preserving location privacy in geosocial applications, IEEE Trans. Mobile Comput. 13 (1) (2014) 159–173.
[27] M. Bellare, A. Boldyreva, A. ONeill, Deterministic and efficiently searchable encryption, in: Advances in Cryptology-CRYPTO 2007, Springer, 2007, pp. 535–552.
[28] G. Yang, C.H. Tan, Q. Huang, D.S. Wong, Probabilistic public key encryption with equality test, in: Topics in Cryptology-CT-RSA 2010, Springer, 2010, pp. 119–131.
[29] Q. Tang, Towards public key encryption scheme supporting equality test with fine-grained authorization, in: Information Security and Privacy, Springer, 2011, pp. 389–406.
[30] R. Fagin, M. Naor, P. Winkler, Comparing information without leaking it, Commun. ACM 39 (5) (1996) 77–85.
[31] M.J. Freedman, K. Nissim, B. Pinkas, Efficient private matching and set intersection, in: Advances in Cryptology-EUROCRYPT 2004, Springer, 2004, pp. 1–19.
[32] L. Kissner, D. Song, Privacy-preserving set operations, in: Advances in Cryptology–CRYPTO 2005, Springer, 2005, pp. 241–257.
[33] E. Novak, Q. Li, Near-pri: Private, proximity based location sharing, in: Proceedings of IEEE Conference on Computer Communications, INFOCOM, IEEE, 2014, pp. 37–45.
[34] G. Saldamli, R. Chow, H. Jin, B. Knijnenburg, Private proximity testing with an untrusted server, in: Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks, ACM, 2013, pp. 113–118.
[35] S. Mascetti, L. Bertolaja, C. Bettini, A practical location privacy attack in proximity services, in: Proceedings of IEEE 14th International Conference on Mobile Data Management (MDM), vol. 1, IEEE, 2013, pp. 87–96.
[36] D.P. Jablon, Strong password-only authenticated key exchange, ACM SIGCOMM Comput. Commun. Rev. 26 (5) (1996) 5–26.
[37] Y. Zheng, M. Li, W. Lou, Y.T. Hou, Sharp: Private proximity test and secure handshake with cheat-proof location tags, in: Computer Security–ESORICS 2012, Springer, 2012, pp. 361–378.
[38] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory 22 (6) (1976) 644–654.
[39] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R.P. Dick, Z.M. Mao, L. Yang, Accurate online power estimation and automatic battery behavior based power model generation for smartphones, in: Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, ACM, 2010, pp. 105–114.
[40] R. Cramer, R. Gennaro, B. Schoenmakers, A secure and optimally efficient multi-authority election scheme, Eur. Trans. Telecommun. 8 (5) (1997) 481–490.