International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 23 Issue 9 – DECEMBER 2016.

A REINFORCED LOCATION PRIVACY SYSTEM WITH DISTRIBUTED AND VENUE CENTRIC MODEL

N. Rahul[#1], A. Vikas [#2] and T. Balachander [*3]

[#] *Student, Department of Computer science and engineering, SRM University, Chennai, India.*

[*] *Assistant Professor, Department of Computer science and engineering, SRM University, Chennai, India.*

***Abstract*— This paper is to propose a new efficient reinforced model for Privacy preserving issue in Location based services. In the existing model the system used a centralized architecture model on which an entity with trusted anonymity server is deployed and used, which was named as Certificate Authority. The approach fails on few aspects and also makes a serious security issue and efficiency drop. To overcome and mitigate the issue in this model we introduce a reinforced model based on similar structure of Peer-to-Peer System model. The trust model provides the user to protect their location information without transmitting those details to anonymous server or authority. A user location is checked and given proof in the Spotter protocol model. So a malicious/attacker user cannot have a trusted connection with the System. The implementation results prove that the proposed model is more efficient and secure by providing both a venue centric and a decentralized solution to the LBS model.**

Index Terms*— Location based service, location proof, location privacy.

I. INTRODUCTION

A Location-Based Service (LBS) takes advantage of the position of its users to deliver a service tailored to their current or past geolocated context. In practice, the position that a user transmits to an LBS is often computed determined by his own device. Thus, a malicious user can lie about his position by having his device transmitting a location of his choice. This type of attack can have a severe impact on applications such as real-time traffic monitoring, location-based access control, discount tied to the visit of a particular shop or local electronic election, to name a few. To counter this threat, LBS should require its users to prove their actual or past position before granting them access to resources. This notion has been formalized through the concept of location proof (LP), which is a digital certificate attesting the position of a user at a specific moment in time. A location proof architecture is a trusted architecture that users can interact with to acquire LPs in a secure manner.

There are many kinds of location-sensitive applications. One category is location-based access control. For example, a hospital may allow patient information access only when doctors or nurses can prove that they are in a particular room of the hospital [19]. Another class of location-sensitive applications require users to provide past location proofs [26], such as auto insurance quote in which auto insurance companies offer discounts to drivers who can prove that they take safe routes during their daily commutes, police investigations in which detectives are interested in finding out if a person was at a murder scene at some time, and location-based social networking in which a user can ask for a location proof from the service requester and accepts the request only if the sender is able to present a valid location proof. The common theme across these location sensitive applications is that they offer a reward or benefit to users located in a certain geographical location at a certain time. Thus, users have the incentive to cheat on their locations.

However, relying on a dedicated architecture to certify the position of users raises important privacy concerns. First, the location privacy of users can be breached due to their regular interactions with the infrastructure (traceability issue). Another issue is for a malicious user to collect proof on behalf of another user with whom he colludes. This problem is known as the terrorist fraud in the literature of distance bounding protocols. Furthermore, since often a LP is actually mainly a time stamped signature of a position, there is no mean for a user to change the granularity of the position endorsed by a LP without risking tampering with its integrity. This property would be particularly interesting with respect to data minimization as it would enable a user to reveal only the granularity of his position (street, district, town, ...) needed by the LBS to ensure its functionality. Finally, in most of the current architectures [16], [27], [24], LPs are stored on centralized servers (although sometimes encrypted [19]) resulting in users losing control of their own location data. As a result, their location privacy might be compromised by hackers or simply abused by the LBS provider. In addition, the replication of storage servers is classically used to ensure the reliability of the system, but this increases at the same time both the risk of leakage and the deployment cost.

Location-sensitive applications require users to prove that they really are (or were) at the claimed locations. Although most mobile users have devices capable of discovering their locations, some users may cheat on their locations and there is a lack of secure mechanism to provide their current or past locations to applications and services. One possible solution

[15] is to build a trusted computing module on each mobile device to make sure trusted GPS data is generated and transmitted. For example, Lenders et al. [15] proposed such a solution which can be used to generate unforgettable geotags for mobile content such as photos and video; however, it relies on the expensive trusted computing module on mobile devices to generate proofs. Although cellular service providers have tracking services that can help verify the locations of mobile users in real time, the accuracy is not good enough and the location history cannot be verified. Recently, several systems have been designed to let end users prove their locations through WiFi infrastructures. For example, Saroiu and Wolman [26] proposed a solution suitable for third-party attestation, but it relies on PKI and the wide deployment of WiFi infrastructure.

A. CONTRIBUTION AND PLAN OF THIS PAPER

In this paper, we propose Reinforced Location Privacy System(RLPS), which does not rely on the wide deployment of network infrastructure or the expensive trusted computing module. In RLPS, Bluetooth enabled mobile devices in range mutually generate location proofs, which are uploaded to an untrusted location proof server that can verify the trust level of each location proof. An authorized verifier can query and retrieve location proofs from the server. Moreover, our location proof system guarantees user location privacy from every party. More specifically, we use statistically updated pseudonyms at each mobile device to protectlocation privacy from each other, and from the untrusted location proof server. We develop a user-centric location privacy model in which individual users evaluate their location privacy levels in real time and decide whether and when to accept a location proof request. In order to defend against colluding attacks, we also present betweenness rankingbased and correlation clustering-based approaches for outlier detection. Extensive experimental and simulation results based on multiple data sets show that RLPS can effectively provide location proofs, significantly preserve the source location privacy, and effectively detect colluding attacks.

II. LITERATURE SURVEY

Most previous works have been focusing on privacy in location queries, i.e., a model in which users report their “encrypted” location data to a central database server to perform range or k nearest neighbour (KNN) queries [11], [12]. Note that in this model the database stored in the server is assumed to be public. In contrast, the recently emerged proximity test is a different model where location-based matching is done only between users, while the users’ locations are private information. Here we briefly describe the concepts of proximity test and private matching.

Proximity test is a special form of location sharing [6], where the information being shared is whether or not two users are within a certain range orin the same geographic region. The main privacy concern in proximity test is that user’s actual location may be involuntarily revealed to either the server or other users. To this end, a privacy-preserving proximity test solution is proposed in [8], using a grid-based encryption algorithm. In [13], Rasmussen et al. devised a privacy-preserving distance bounding protocol, using stream cipher. However, their protocol relies on a pre-shared secret key to initialize, and is subject to dictionary attack [14], [15]. In [16], Mascetti et al. proposed proximity detection schemes based on service provider filtering, in which privacy protection is achieved by user-chosen location representation that controls its granularity. However, their protocol leaks coarse-grained location information to the server. In [1], Narayanan et al. proposed a suite of private proximity test protocols. The possibility of constructing location tags from environmental signals was noted; however, their protocols either require a pre-shared secret key between users, or are not scalable and efficient enough to handle one-to-many proximity test as studied in our paper. Another proximity test scheme was proposed in [9], where users can also control their privacy levels via leveled publishing. The protocol is based on keyed hashing which suffers from dictionary attacks. In [17], Lin et al. proposed a proximity test scheme by applying shingling technique [18] to GSM cellular messages. However, they did not thoroughly analyze its security. In our previous work [19], we designed a two-step private proximity test protocol using unforgettable location tags. However, the original protocol cannot detect fine-grained location cheating in the second step. In this paper, we provide an improved protocol that can be robust against fine-grained location cheating using an additional peer-to-peer location proof mechanism. We carry out a systematic study of unforgettable location tags, and their usage in proximity test based on a more thorough analysis, realistic simulations, and more comprehensive real-world experiments.

III. PROBLEM STATEMENT

A LP is generated from a collection of LPSs, and can be manipulated to disclose the location information at different levels of granularity. A user corresponds to an entity using the location proof system. It typically refers to both the device carried out by an individual and the individual himself.

A user can take one or several of the following roles: prover, witness or verifier. A prover is a mobile user of the system that periodically collects LPs. In contrast to some previous works, we do not assume that the prover is required to know in advance the LBS he will interact with when collecting LPs. The prover stores all LPs collected on a personal device (e.g., his smartphone) under his control to use them at a later time. A witness is located in the vicinity of a prover and accepts to participate to the generation of a LPS for this prover. The identity of a witness must be kept secret from the other users of the system, with the exception of the Anonymity Lifter (defined later). A verifier checks the validity of the position claimed by a prover through a (e.g., bank, store, social networking site, police authority, LBS provider) or another user.

The Certification Authority (CA) is a trusted third party responsible for issuing the credentials to newly registered users. These credentials can be considered as being the “identity” of these users. This authority is only used to register new users and is not involved in the generation of LP.

The Anonymity Lifter (AL) is a trusted third party that has

the capacity to lift the anonymity of a particular user when needed (for instance upon request from a judge). Users are mobile entities capable of positioning themselves into space. For instance, a user might be able to localize himself by using his GPS or through the help of a dedicated positioning infrastructure. We assume the imprecision of the location computed to be negligible.Each user possesses a local clock on his device that is synchronized with the clocks of other users. Thus, when two users communicate together, they rely on the same time referential. In practice, users can use the clocks of their GPS or GSM device.

Provers can broadcast message to neighboring witnesses without disclosing identifying information (e.g., MAC or IP addresses). C. Adversary Models When reasoning about the security and privacy of the architecture, we will consider the following types of adversary. Each of these adversaries is assumed to be computationallybounded (i.e., he cannot break the cryptographic assumptions on which the security of cryptographic primitives rest).

This adversary has the capacity to wiretap on the communications exchanged between users in his vicinity. His main objective is to break their anonymity. A malicious prover aims at obtaining LPs without physically being present at a location. For instance, he can try to modify the time and position information endorsed in a LPS already issued by a witness, or lie to a verifier about the position enclosed in a LP.

A malicious verifier may want to extract the identity of the prover out of one of his LP, to breach his location privacy by obtaining more precise information than the one disclosed by the prover or even impersonate him in front of another verifier by pretending to be the owner of the LP. This witness may fool an honest prover by endorsing different spatio-temporal information than the one requested by the prover or collude with a prover by generating more than one LPS for him.

IV. MOTIVATION

The main security goal for proximity test is to design an unforgettable location proof so that the protocol is robust against location cheating. Location cheating happens when one party is able to deceive the other party with an untruthful location. In our case, if Bob can trick Alice into believing that he is within her vicinity region while he actually is not, he has successfully launched a location cheating attack. Unforgettable location proofs are extremely important for location based services. To the best of our knowledge, we are among the first to address location unforgettability in proximity test.

The privacy goal of the protocol is to maintain each user's location privacy. Specifically, through the proximity test, the server cannot learn any users' locations. Alice should only learn the identities of the candidates who are within her vicinity region but not the exact locations of those candidates. Alice should not learn anything about the candidates who are outside of her vicinity region. The candidates should not learn whether they pass Alice's proximity test or anything that reveals Alice's location or the location of her vicinity region.

Existing private proximity test protocols operate on pairs of users. If Alice wants to test a group of N users, she has to run the protocol N times with every user in the group. This results in a bandwidth complexity of OpNq and a computational complexity of OpNq at Alice's side. Our goal is to design an efficient protocol where Alice and each participant only submit their information once to the server. This leads to a communication complexity of Op1q at the user's side. This represents a significant efficiency improvement compared to the existing schemes.

V. PROPOSED SYSTEM

The proposed system explored two possible sources of location tags: (1) Using 802.11 MAC headers in WiFi network. MAC frame headers are a family of 34-bit vectors that contain control, duration, address, and sequence control information of the frame. In our assessment, we consider the most common 802.11 MAC headers as shown in Fig. 1A. (2) Using radio network temporary identifiers (RNTIs) in LTE networks. RNTIs are a family of 16-bit vectors that are used to differentiate a radio channel or a user from others. There are various types of RNTIs being used in a LTE network. In Fig. 1B, we list the ones that can be captured by common mobile devices. Both 802.11 MAC headers and LTE RNTIs are time-varying and location-specific, i.e., users who are observing similar sets of 802.11 MAC headers or LTE RNTIs are likely within the same region at the same time. There are two additional reasons for our choices. First, they are the result of a compromise between unpredictability and reproducibility. In our early design, we experimented using full frames as location tag sources. The resulting location tags contain abundant entropy and are highly unpredictable. However, they are difficult to reproduce even

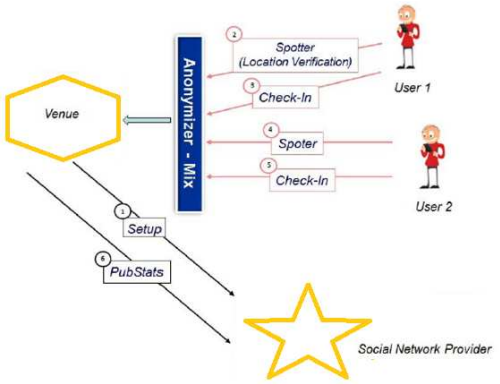

Figure 1. Architecture of the proposed system

A location tag can be regarded as a token of proof associated with a point in space and time. It is collections of signals presented at a certain location at a certain time. Let X be a set of environmental signals captured at a point in space and time. A location tag Y is a subset of X , selected by a filter function φ: X Ñ Y Ď X . Each element, y, of Y is an observation of the location tag. From the functionality point of view, a good location tag should at least have the reproducibility property. That is, if two measurements at the same space and time yield tags Y1 and Y2, then Y1 and Y2 match with high probability. On the other hand, from the security point of view, in order to be cheat-proof, a good location tag must have the unpredictability property. That is, an adversary not at a specific place and time is unable to produce a tag that matches the tag constructed at that location at that time. We can quantitatively measure a location tag's

unpredictability through its entropy.

A. CORRECTNESS

The completeness property is trivial. Once an LPS is received by the prover, he can verify that the spatio-temporal information contained within it is valid. The spatial and temporal soundness are ensured because revealing a geolocated context that does not match the one contained in the LP will be detected during the Step 4 of the verification process. Thus, a malicious prover cannot alter the integrity of a LPS and fool the verifier by claiming a different location than the one contained in the LPS. In the following, we give more details about how PROPS ensures the spatial soundness property (cf. Section III) by proving its resistance to the distance fraud and mafia fraud.

B. RESISTANCE TO DISTANCE FRAUD.

In a distance fraud, a malicious prover tries to convince an honest witness that he is closer than in reality. By assumption in PROPS, the distance fraud is prevented by the use of the Proximity Testing(δ, C1) protocol. Let P be an honest prover located at position Lp and W an honest witness located at position Lw such that dist(Lp, Lw) > δ. Consider W^i and P^i, which are two different colluding users or the same malicious user playing two different roles. In the mafia fraud, the objective of an adversary is to replay a session that involved a honest prover P to fool W and make him believe that P is closer than he really is. Thus, two sessions need to be run, the first one involving W^i and P with the objective to get commitments from P followed by a second one performed between P^i and W in which the commitments of the first protocol are replayed in order to obtain a location share on behalf of P. In the second session P^i will need to compute a fresh zero-knowledge proof using r from W to prove knowledge of the identity s_u of P, which is impossible without the knowledge of s_u. Proof of ownership and non-transferability. During the verification phase, the verifier checks that the current prover is effectively the legitimate owner of the LP by running a zero-knowledge protocol over the pseudonym C1 included in the proof. Within PROPS, the non-transferability property is equivalent to the resistance to the collusion P − P. The resilience to the collusion P − P follows directly from the resilience to the terrorist fraud of the DB protocol used.

C. UNFORGEABILITY

The unforgeability is ensured partially by the uniqueness property provided by unique group signature, which prevents the adversary controlling a collusion of m malicious users to gather enough LPSs as long as the size of the collusion is less than the number of shares needed (i.e., k>m).

D. RESISTANCE TO DISTANCE HIJACKING.

In a distance hijacking attack, a malicious user M tries to hijack the gathering session of an honest prover P. More precisely, M waits until P has successfully proved that he is in the vicinity of an honest witness W and then hijacks P's session to collect its LPS. However in PROPS, a witness verifies that the entity who ran ProximityTesting(δ, C1), is the same as the one that computes the commitment C1. Therefore, such an attack will be detected by W and the gathering process will be aborted before the malicious prover receives the LPS, thus avoiding the possibility of hijacking.

E. ANONYMITY OF PROVER AND WITNESSES

Due to the use of commitments and zero-knowledge proofs in RLPS, users can remain anonymous in the system as long as they behave honestly. Furthermore, the prover creates periodically nonces to generate new pseudonyms. Therefore, the pseudonyms are unlinkable provided the nonces are chosen at random and independently. Moreover due to the hiding property of the commitments, the pseudonyms do not disclose any information that can be used to trace back to the identity of the prover. In addition, the anonymity and unlinkability of witnesses are ensured by the use of group signature. Finally, the use of a zero-knowledge proof enables the prover to anonymously authenticate himself to the verifier as the owner of a LP.

F. WITNESS LOCATION PRIVACY

When establishing a LP, a witness never discloses his exact position but rather checks that the position claimed by the prover is in the vicinity using the proximity testing protocol. Therefore, a local eavesdropper can only infer that the witness is in the proximity of the prover but does not learn his exact position. The location information is first encoded into the hash chains before being endorsed by the witness. This information cannot be modified later by the prover and it does not appear in clear in the LP.

G. LOCATION SOVEREIGNTY

Within PROPS, the LPs gathered by a user are saved locally on his device in contrast with other schemes in which the proofs are stored and controlled by remote servers. Finally, due to the use of hash chains the prover can decide the granularity of the information he wants to disclose.

VI. IMPLEMENTATION

Group signature schemerelies on a single public verification key for the group but a different private key for each signer (i.e., member of the group). Each member of the group can issue a signature on a message using his private key and the authenticity of the signature is checked using the group verification key. Thus, for any computationally-bounded adversary, it is impossible to identify the actual signer of a message. The main operations of a group signature scheme with optional anonymity lifting are the following ones.

Init(1λ). This procedure generates the parameters of the group (λ is a security parameter). It outputs the following keys: gpk the public verification key of the group signature, ok the opening key needed to lift the anonymity of a user and ik the issuing key needed to dynamically add user to the group of signers.

Join(useri, ik). This procedure takes as input the issuing key ik and a user's identity useri. At the end of the procedure, useri receives gsk[i], his private group signature key and becomes officially a member of the group.

GroupSign(m, gsk[i]). This procedure takes as input a

message m and a signature key gsk[i], and then produces a group signature σi of the message m.

GroupVerif(m, σi, gpk). This operation enables to check the authenticity of a group signature. It requires as input the group verification key gpk, a message m and a group signature on this message σi. GroupVerif returns either accept or reject depending on the validity of the signature.

LiftAnonymity(m, σi, ok). This procedure retrieves the identity of a particular signer from a signature σi. This operation takes as input a message m, a group signature on this message σi, the opening key ok and produces as output the identity useri of the signer. As explained above, group signature schemes are designed to ensure the anonymity to a signer of the message and multipleshow unlinkability. Indeed, it is impossible to distinguish if two signatures originate from the same member. In the context of location proof system, we would like to avoid potential abuses such as the unforgettability property and the possibility for one witness to provide more than one LPS per request of a prover. This issue can be solved by relying on unique group signature scheme. More precisely in a unique group signature scheme, if a signer produces two signatures on the same message (i.e., two LPSs for the same location information), then there exists an efficient algorithm to detect this:

Detect(m, σ, σ). This procedure is a detection algorithm that can tell if two signatures σ and σ are signatures on the same message m by the same user. If this situation occurs, this procedure returns true while otherwise it returns false. Unique group signature scheme can be implemented for instance using CL-signature. This efficient group signature scheme relies on bilinear maps and its security is solely based on the LRSW assumptions.

A commitment scheme is a cryptographic primitive enabling a prover to hide a value of his choice such that he can decide later to reveal it. In a nutshell, a commitment scheme consists of two algorithms. First, the Commit algorithm takes as input a value m and a random string r, and then outputs a commitment C. Second, the opening algorithm VerifyCommit takes as input C, m and r, and then outputs accept if C = Commit(m, r) and reject otherwise. A cryptographically secure commitment scheme is hiding in the sense that it is computationally hard to infer m given C and binding meaning that it is hard to find m = m such that C = Commit(m , r).

The Pedersen commitment works in the following manner. Given a group G of prime order q with generators g and h, a commitment to x ∈ Zq is formed by choosing a random r ∈ Zq and setting the commitment C = gx × hr. This commitment scheme is information-theoretically hiding, and is binding under the discrete logarithm assumption, which is actually directly implied by the LRSW assumption.

Zero-knowledge Proof A Zero-knowledge proofis a protocol (interactive or non-interactive) that enables a prover to convince a verifier that he possesses a proof of the veracity of a mathematical statement without leaking any information about the proof itself. Using the notation a zeroknowledge protocol can be written as ZKProof{(w) : F(w) = 1}, in which F denotes a mathematical statement (a language) and w represents a proof of this statement. Zero-knowledge proof can be used to prove various properties such as the knowledge of a discrete logarithm or of quadratic residues.

Over the years, zero-knowledge proofs have been widely used to develop anonymous credentials systems. The most popular of these systems include the Direct Anonymous Attestation (DAA) protocol, the identity mixer anonymous credential system and the CL-signature. In our context, we are interested in a zero-knowledge proof to convince that two Pedersen commitments are commitments to the same value. We call this procedure EqualityCommitment, and it is run interactively between the prover and a witness.

In the following, we give the intuition of how this procedure works. Without loss of generality, consider C1 = gx × hr1 and C2 = gx × hr2 the commitments that the prover needs to convince that they correspond the same value x. The prover picks randomly ρ1, ρ2 ∈ Zq and computes ω = gρ1 × hρ2 before sending it to the witness. The witness chooses a challenge e ∈ Zq and forwards it to the prover. Then, the prover sends back s1 = ρ1 + e × x, s2 = ρ2 + e × r1 and s3 = ρ2+e×r2. If gs1×hs2 = ω×Ce 1 and gs1×hs3 = ω×Ce 2 , then the procedure EqualityCommitment returns accept and the witness accepts the proof, otherwise it returns reject and the witness rejects the proof.

In RLPS, we assume the availability of a proximity testing procedure that is used by a prover to convince a witness that he is close to him. The success of this procedure is a prerequisite to the issuance of a LPS using a valid pseudonym. In practice in PROPS, the pseudonym is a commitment over the long term secret of the prover. A proximity testing procedure from distance-bounding (DB) protocols can easily be adapted for our needs but our architecture is actually agnostic to the DB protocol used. In the following, we describe the execution of the proximity testing procedure that is inspired from the Bussard-Bagga protocol.

The proximity testing procedure consists of three phases. The first phase is the preparation one, in which the prover encrypts his private key SU with a random symmetric key k and gets the corresponding encrypted message e. Then, the prover commits individually to each bit of e and k, which results in two sequences of bit commitments R0 and R1. This phase can be performed offline by the prover to save time. During the second phase, the prover sends R0 and R1 to the witness, which then starts a multi-round fast-bit-exchange with the prover. In each round i, the witness sends a challenge bit bi ∈ {0, 1}, to which the prover replies with the i-th bit of Rb. Since the witness never learns both bit values, he will also never learn the secret SU . After the multi-round fast-bit-exchange, the witness verifies the corresponding bit commitments of R0 and R1 (only for the received bits) by asking the prover to provide the opening information for these commitments.

During the third phase, the values R0 and R1 are used by the witness to derive a pseudonym C2. Finally, the prover convinces the witness that C2 and C1 correspond to Pedersen commitments on the same value through a zero-knowledge proof. For more details about how the pseudonym C2 is constructed, we refer the reader to. In contrast to the original protocol from and its implementation in STAMP, the witness and the verifier do not need to have a public key to authenticate the prover. Instead the authentication is performed using pseudonyms and zero-knowledge proofs to

preserve the privacy of the prover. Revealing the values k and e to a colluder also disclose the long-term secret of the prover, thus ensuring that the proximity testing procedure is resistant to terrorist frauds. Thereafter, we denote the proximity testing procedure by ProximityTesting(δ, C1), in which δ represents the distance threshold used to verify the proximity of the prover and C1 is the current pseudonym used by the prover at the initialization of the protocol.

As mentioned previously, the user of a location proof system should have the possibility to reveal different granularities of the positions contained in the LPSs he collected. More precisely when creating an LPS, each witness generates five different granularities of the location of the prover. The granularities are then encrypted with different keys using a symmetric encryption algorithm such as AES.

The encrypted values are then endorsed by the witness and put in the LPS. When a prover reveals his location up to a particular granularity to a verifier, he simply sends the decryption key corresponding to the granularity he wishes to disclose (this method is used for instance in STAMP). In this approach, the digits of a GPS coordinates are hidden into hash chains. Then, revealing the leftmost digits can generate a new granularity of the position. Without loss of generality, we assume that each GPS coordinates pos, such as latitude and longitude, is represented by d digits in decimal (i.e., xi = xi dxi d−1 ...xi 1). The hash chain is composed of the following operations.

Hide(pos, seed). This procedure is run by the prover and takes as input his precise position pos, a secret string seed and outputs Kpos, which is the encoding of pos under the secret string seed. In practice, Kpos is the last value of a hash chain and corresponds to the information that will be certified by the witness and included in the LPS instead of pos.

Reveal(pos, p, seed). This procedure is called by the prover in order to partially reveal his position at granularity p of his choice. Reveal takes as input the previous position pos, a granularity p such that $p \leq d$ and the same secret seed used to encode pos with the algorithm Hide(pos, seed) and outputs the pair (Lpos, auxpos) in which Lpos represents the position pos revealed up to the granularity p and auxpos is an auxiliary information needed to prove that Lpos is well formed.

Check(Kpos, auxpos, Lpos). This procedure is used by the verifier to check that the pair (Lpos, auxpos) revealed by the prover corresponds to the location information contained in Kpos. This procedure takes as input Kpos, auxpos and Lpos and returns accept if the location information claimed by the prover is verified, and reject otherwise. To encode the temporal information, the prover format the current time into five values (i.e., {x 1,...,x 5}) that correspond to the time (hh:mm:ss), period of the day (morning, afternoon or night), day, month and year. Then, he also relies on a hash chain to encode the temporal information in the LPS. A GPS coordinate relies on seven digits for the precision, thus the prover can hide a maximum of six digits.

VII. CONCLUSION

In this paper, we introduced RLPS, a novel privacypreserving location proof system based on a collaborative architecture. The main strengths of this location proof system are the following: (1) the LP collected by a prover are under his control and does not reveal any information about his identity, (2) the prover has the ability to remain anonymous even when presenting a proof to a verifier, (3) the privacy of users is preserved with respect to a man-in-the-middle adversary, (4) a verifier can detect abuses of a malicious user that tries to produce fake LPs, and finally (5) the prover can selective disclose the spatial and temporal information to the granularity of his choice..

In the future, we would like to extend RLPS to deal with the collusion in which a witness systematically reports false LPS for a colluder even though one or both of them are not at the location claimed in the LPS. Indeed, unless trusted infrastructures are deployed at each possible location, it seems quite difficult to detect that a particular LPS is a result of such collusion. A line of research that we would like to pursue in the future is the use of anonymous peer-to-peer reputation system as a countermeasure to frauds in this mobile environment. Finally, another research avenue is the design of a secure multiparty computation version of the protocol involving a joint interaction with the prover and multiple witnesses rather than relying on pair wise interactions between the prover and each witness.

REFERENCES

[1] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, “Location privacy via private proximity testing,” in Proc. 18th Annual Network & Distributed System Security Symposium, 2011.

[2] M. Li, N. Cao, S. Yu, and W. Lou, “Findu: Privacy-preserving personal profile matching in mobile social networks,” in Proc. 30th IEEE International Conference on Computer Communications, 2011.

[3] M. Li, W. Lou, and K. Ren, “Data security and privacy in wireless body area networks,” Journal of Wireless Communications, vol. 17, no. 1, 2010.

[4] X. Liang, R. Lu, L. Chen, X. Lin, and X. Shen, “Pec: A privacypreserving emergency call scheme for mobile healthcare social networks,” Journal of Communications and Networks, vol. 13, no. 2, 2011.

[5] W. He, X. Liu, and M. Ren, “Location cheating: A security challenge to location-based social network services,” in Proc. 31st IEEE International Conference on Distributed Computing Systems, 2011.

[6] J. Y. Tsai, P. G. Kelley, L. F. Cranor, and N. Sadeh, “Location-sharing technologies: Privacy risks and controls,” I/S: A Journal of Law & Policy for the Information Society, vol. 6, pp. 119–317, 2010.

[7] Z. Zhu and G. Cao, “Applaus: A privacy-preserving location proof updating system for location-based services,” in Proc. 30th IEEE International Conference on Computer Communications, 2011.

[8] L. Sikˇ snys, J. R. Thomsen, S. ˇ Saltenis, M. L. Yiu, and O. Andersen, ˇ “A location privacy aware friend locator,” Advances in Spatial and Temporal Databases, pp. 405–410, 2009.

[9] L. Sikˇ snys, J. R. Thomsen, S. ˇ Saltenis, and M. L. Yiu, “Private and ˇ flexible proximity detection in mobile social networks,” in Proc. 11th IEEE International Conference on Mobile Data Management, 2010.

[10] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, “Fuzzy extractors: How to generate strong keys from biometrics and other noisy data,” in Advances in Cryptology-Eurocrypt 2004. Springer, 2004, pp. 523–540.

[11] J. Meyerowitz and R. R. Choudhury, “Hiding stars with fireworks: Location privacy through camouflage,” in Proc. 15th ACM Annual International Conference on Mobile Computing and Networking, 2009.

[12] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J.C. Herrera, A.M. Bayen, M. Annavaram, and Q. Jacobson, “Virtual Trip Lines for Distributed Privacy-Preserving Traffic Monitoring,” Proc. ACM MobiSys, 2008.

[13] T. Jiang, H.J. Wang, and Y.-C. Hu, "Location Privacy in Wireless Networks," Proc. ACM MobiSys, 2007.
[14] V. Kostakos, "Experiences with Urban Deployment of Bluetooth," presentation given at the Univ. of California, San Diego, Mar. 2007.
[15] V. Lenders, E. Koukoumidis, P. Zhang, and M. Martonosi, "Location-Based Trust for Mobile User-Generated Content: Applications Challenges and Implementations," Proc. Ninth Workshop Mobile Computing Systems and Applications, 2008.
[16] M. Li, R. Poovendran, K. Sampigethaya, and L. Huang, "Caravan: Providing Location Privacy for VANET," Proc. Embedded Security in Cars (ESCAR) Workshop, 2005.
[17] M. Li, K. Sampigethaya, L. Huang, and R. Poovendran, "Swing & Swap: User-Centric Approaches Towards Maximizing Location Privacy," Proc. Fifth ACM Workshop Privacy in Electronic Soc., 2006.
[18] Y. Li and J. Ren, "Source-Location Privacy Through Dynamic Routing in Wireless Sensor Networks," Proc. IEEE INFOCOM, 2010.
[19] W. Luo and U. Hengartner, "Proving Your Location Without Giving Up Your Privacy," Proc. ACM 11th Workshop Mobile Computing Systems and Applications (HotMobile '10), 2010.
[20] J. Manweiler, R. Scudellari, Z. Cancio, and L.P. Cox, "We Saw Each Other on the Subway: Secure Anonymous Proximity-Based Missed Connections," Proc. ACM 10th Workshop Mobile Computing Systems and Applications (HotMobile '09), 2009.
[21] J. Manweiler, R. Scudellari, and L.P. Cox, "SMILE: EncounterBased Trust for Mobile Social Services," Proc. ACM Conf. Computer and Comm. Security (CCS), 2009.
[22] F.J. Massey Jr., "The Kolmogorov-Smirnov Test for Goodness of Fit," J. Am. Statistical Assoc., vol. 46, no. 253, pp. 68-78, 1951.
[23] A. Pfitzmann and M. Hansen, "Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management—A Consolidated Proposal for Terminology" (Version v0.31 Feb. 15, 2008), http://dud.inf.tu-dresden.de/literatur, 2008.
[24] I. Rhee, M. Shin, K. Lee, and S. Chong, "On the Levy-Walk Nature of Human Mobility," Proc. IEEE INFOCOM, 2007.
[25] J.L. Romeu, "Kolmogorov-Simirnov: A Goodness of Fit Test for Small Samples," START: Selected Topics in Assurance Related Technologies, 2003.
[26] S. Saroiu and A. Wolman, "Enabling New Mobile Applications with Location Proofs," Proc. ACM 10th Workshop Mobile Computing Systems and Applications (HotMobile '09), 2009.
[27] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards Statistically Strong Source Anonymity for Sensor Networks," Proc. IEEE INFOCOM, 2008.
[28] N. Smart et al., "Ecrypt II Yearly Report on Algorithms and Key Lengths (2009-2010)," ICT-2007-216676, Revision 1.0, 2010.