Security Analysis of The Louis Protocol for Location Privacy

Aakar Gupta, Milan Saini and Anish Mathuria
Dhirubhai Ambani Institute of Information and Communication Technology
Gandhinagar, Gujarat, India 382007
Email: {aakar_gupta, milan_saini, anish_mathuria}@daiict.ac.in

***Abstract*—Many location-based services for alerting persons of nearby friends have been deployed in practice. A drawback of most approaches to providing such services is that friends always learn each other's location even when they are not actually nearby. The Louis protocol proposed by Zhong, Goldberg and Hengartner aims to ensure that a friend's location is revealed to another friend if and only if the friends are actually nearby. The protocol lets a third party learn whether the friends are nearby, without the third party learning their location. The third party communicates the answer to the person who invokes the service. A key feature of the protocol is that a person can detect misbehavior by the third party or the person's friend. This paper reveals a flaw in the way the protocol handles the detection of the misbehaving party, leading to an unauthorized disclosure of a person's location. Two alternatives for fixing the flaw in the protocol are proposed and a heuristic analysis is given.**

I. Introduction

Gone are the days when the only requirement met by a mobile phone was to allow people to talk to each other. Modern mobile devices provide lot more than this basic facility. One such value added service category is that of Location Based Services (LBS). According to Virrantaus, Markkula, Garmash and Terziyan [16], "LBSs are information services accessible with mobile devices through the mobile network and utilizing the ability to make use of the location of the mobile device." Simply put, location based services, besides other things, take the geographical position (actual location co-ordinates or relative distance from a reference point) of the user as a necessary input to provide a personalized service, hence the name. Location privacy is the property which states that a user's location attributes (distance from a point or actual co-ordinates) should not be revealed unless it is authorized by the user.

The recent boom in location based services has given rise to a number of Buddy Tracking and friend finding applications [20], [21], [22], [23], [24], [25], [26]. These services notify the users of their friends in the vicinity or allow them to view the locations of their friends on a map. Though each of them lets the user control whom they want their location revealed to and when they want it, the degree of privacy and security offered to the users varies with the service; for example MyLoki [23] and iPling [22] simply allow all the friends whom the user has approved, to get location information without any constraints being put on how far the user and his/her friend are from each other. This results in friends knowing each others' location information without being nearby. Other services [20], [21], [24], [25], [26] send alerts to a user if the user is nearby. But the drawback of these services, other than being specifically dependent on a particular cellphone network architecture and the limited options given to the user for defining the nearby area, is that they all are required to reveal the location information to the service provider. MIT's iFind project [2] introduces a distributed buddy-tracking application, where an individual's WiFi device determines its location and shares this information with his friends. Though this approach can be exploited for alerting people of nearby friends, its drawback is that the friends always learn each other's location, not considering whether they are actually nearby; thus resulting in, possibly, revealing more information than desired. Let us consider a scenario. Alice is at an invitation-only gathering, and wants to find out if her contemporary Bob is there too. If he is, she wants to exchange location and meet up with him; but if not, then she does not want him to know her location, as it could hurt Bob. Thus she needs to know if Bob is nearby and only then can she exchange the location information with him. Consider another scenario where an organization implements such a service to locate its employees while they are within the organization's premises. Here the flaw is evident, that is, because of this service the organization will be able to track the employees wherever they are, thus breaching their personal privacy. So what is actually needed is a service which lets one know of his/her friends' location attributes only when they are nearby. This problem has been explicitly addressed by Zhong, Goldberg and Hengartner [1] and has been referred to as the *nearby-friend problem*. The protocols proposed by Zhong et al. do not require a separate service provider that is aware of user's locations. It makes use of secure multiparty computation (SMC). No other previously proposed protocol directly handles this problem, although other SMC problems such as the point-inclusion problem introduced by Atallah and Du [11], [12] could be used to solve the problem. This is further discussed in Section II.

Zhong et al. proposed three distributed protocols - Louis, Lester and Pierre [1] to solve the nearby friend problem. The Pierre protocol achieves the objective of telling Alice whether Bob is nearby or not, but at the cost of not being able to tell

the user the precise distance to a nearby friend. The Lester protocol tells the precise distance, but has the drawback that a user might be able to learn a friend's location even if the friend is in an area that is no longer considered nearby by the friend. In the Louis protocol, which requires a semi-trusted third party, it is claimed that after the protocol has been run, the two involved parties (principals in the protocol) share their location co-ordinates *if and only if they are actually nearby*. (We will refer to this goal of the protocol as the *only nearby friend* clause in the rest of this paper.) We show that the Louis protocol is unsuccessful in maintaining the nearby friend condition, in some of the cases when a party tries to cheat.

The rest of this paper is organized as follows. In Section II, we describe the related work that has been done in this field. The Louis protocol is then described in Section III. In Section IV, we present a security analysis of the protocol highlighting its flaw. In Section V and VI, we describe proposed solutions intended to overcome the discovered flaw in the protocol along with their security analysis. And finally we conclude the paper by describing possible directions for future work.

II. Related Work

Location cloaking is the approach where the location of a device is hidden (by the device or a third party) before sending it to the service provider of the location based service. It is a popular approach for providing location privacy [4], [5], [6], [7]. Location cloaking is studied by Cheng, Zhang, Bertino and Prabhakar [4] for a service that alerts people of nearby friends. The service provider of LBS knows only that the user (who is requesting the service), is within a particular region, but not the exact location. The quality of the answer received by the user from the service is dependent on the privacy information, the user is willing to trade-off. A drawback of this approach, as has also been pointed out by Zhong et. al. [1] is that the service provider learns some location information. The Louis protocol does not need such a third party, that is, it does not require the active involvement of the service provider. (In the Louis protocol, the third party does not learn any location information.) Furthermore, there is no trade-off between the quality of the answer and the privacy information revealed.

Instead of revealing location attributes to a service provider, the nearby-friend problem uses secure multiparty computation (SMC), where multiple parties jointly compute the output of a function without learning each other's inputs. Many SMC protocols which calculate distance between two nodes, without revealing the individual location co-ordinates to each other have been proposed [8], [9], [10]. But these are not suitable for the nearby friend problem, as they reveal the distance between the nodes to the involved parties, which is a location attribute that has to be kept private in the nearby friend application, until the nodes are proved nearby.

Atallah and Du identified the point-inclusion problem in 2001 as an SMC problem [11], [12]. Bob has a private point and Alice has a private polygon, and they want to find out whether the point is inside the polygon or not. Bob does not want Alice or anyone else to know any information about the point, and likewise, Alice does not want Bob or anyone to know any information about the polygon. The point-inclusion problem directly addresses the requirements of the nearby friend problem, and can be exploited for letting Alice know whether Bob is nearby. Du et al. propose a protocol for this in [12], which lets both Alice and Bob learn whether Bob's point is in Alice's polygon. As has been pointed out by Zhong et al. [1], the resource usage and the number of steps required for this protocol is very high. The Louis protocol, which needs a semi-trusted third party, lets Alice know in four communication steps whether Bob is nearby and requires one additional step to inform Bob of this result.

Køien and Oleshchuk [15] also propose a secure two-party protocol for the point-inclusion problem. However, as pointed by Zhong et al. [1] the protocol has a flaw: Alice can learn Bob's location by choosing a degenerate polygon. For example, if there are only two different edges in the polygon and all the other edges are identical to one of them, Alice will usually be able to solve a system of linear equations to determine Bob's location. Bob cannot detect degenerate polygons, assuming the underlying encryption scheme is semantically secure, so this protocol is not adequate for solving the nearby-friend problem. Thomas [17] considers the point inclusion problem in a star-shaped domain and a more general polygonal domain (can have several disconnected nested components). But, it is susceptible to a similar attack as that on Køien and Oleshchuk.

Li and Dai [9] studied the point-inclusion problem for circular domain. However, their solution is not secure in the sense that each party gets additional information regarding the location of the other party's object, that is, distance. Moreover, their solution is highly inefficient. A more efficient protocol for point-inclusion, which uses homomorphic encryption [12], [13], [19], [14] in a circular domain was proposed by Luo, Huang and Zhong [10]. This protocol is the closest in methodology to the Louis protocol, but the problem with this protocol is that the output is revealed to only one of the parties, the other party has to trust the knowledgeable one for the output. In the Louis protocol, after Alice learns that Bob is nearby she can prove this to Bob.

III. The Louis Protocol

The Louis protocol involves three principals: Alice (the initiator), Bob (the responder) and Trent (the semi-trusted third party). As shown in Fig. 1, Alice exchanges messages with Trent and Bob. Bob and Trent do not exchange messages

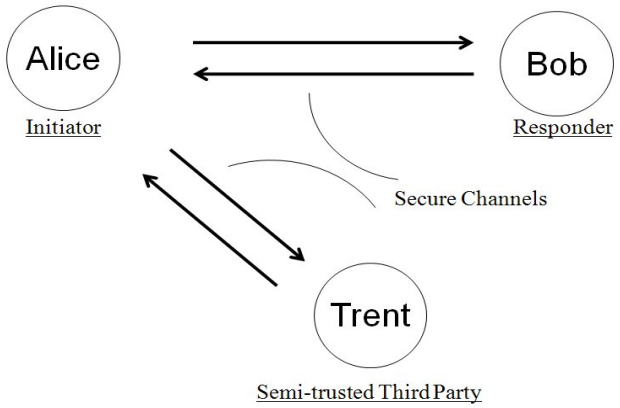

Fig. 1. Setup of the Louis Protocol

with each other. Therefore Trent never learns about Bob's identity. Trent learns whether two friends are nearby, without learning their location attributes and is hence, termed as semi-trusted. The protocol is designed to operate even when each party acts maliciously, although it does not handle collusion among the parties.

Alice and Bob are friends who want to find out whether they are nearby. It is assumed that location of a principal can be mapped to two-dimensional co-ordinates and Alice and Bob understand this mapping. Alice considers Bob to be nearby if he is within a circle of radius r (decided by Alice, during the protocol run) centered around Alice. This value of r must not be revealed to Trent. Two secure communication channels (as shown in Fig. 1) are setup before the protocol begins (one between Alice and Bob and the other one between Alice and Trent). Each channel provides confidential and authenticated communication between the two communicating principals.

A. Notations

- (x, y) denotes Alice's cartesian location co-ordinates and (u, v) denotes Bob's co-ordinates.
- r is the radius of the circle centered around Alice.
- D is the distance between Alice and Bob, $D = \sqrt{(x-u)^2 + (y-v)^2}$.
- d is the difference of squares of distance and radius, that is, $d = D^2 - r^2 = (x-u)^2 + (y-v)^2 - r^2$.
- $\varepsilon_A(\cdot)$ denotes the Paillier encryption using Alice's public key.
- $\varepsilon_T(\cdot)$ denotes non-homomorphic encryption using Trent's public key.
- $H(\cdot)$ is a cryptographic hash function.
- $sig_X(m)$ is a non-message recovering signature on message m by principal X.
- k is a random value chosen by Bob.
- s_X is a random salt chosen by principal X.

B. Paillier cryptosystem

The Paillier cryptosystem [3] is an encryption technique where one can perform a specific algebraic operation on the plaintext by performing an (possibly different) algebraic operation on the ciphertext. If the algebraic operation being performed on the plaintexts is addition, then the technique is called additive homomorphic encryption. In other words, given $\varepsilon(m_1)$ and $\varepsilon(m_2)$, one can efficiently compute $\varepsilon(m_1 + m_2)$. In Louis protocol this technique enables one of the principals to perform a computation without actually knowing the values.

Alice selects random primes p and q and computes $n = pq$. Plaintext messages are elements of Z_n and ciphertexts are elements of Z_{n^2}. Alice picks up a random $g \in Z^*_{n^2}$ such that $\mu = (L(g^\lambda \ mod \ n^2))^{-1} \ mod \ n$ exists, where $\lambda = lcm(p-1, q-1)$ and $L(x) = (x-1)/n$. Alice then uses (n, g) as her public key and (λ, μ) as her private key.

1) Encryption: In order to encrypt a message m, Bob, computes the ciphertext $E(m) = c = \varepsilon(m) = g^m \cdot \gamma^n \ mod \ n^2$, where $\gamma \in Z^*_n$ is a random number.

2) Decryption: In order to decrypt the encrypted message, Alice computes $D(c) = (L(c^\lambda \ mod \ n^2)) \cdot \mu \ mod \ n$, which reveals m to her. Given two ciphertexts, $\varepsilon(m_1) = g^{m_1} \cdot \gamma_1^n \ mod \ n^2$ and $\varepsilon(m_2) = g^{m_2} \cdot \gamma_2^n \ mod \ n^2$ one can efficiently compute $\varepsilon(m_1 + m_2) = \varepsilon(m_1) \cdot \varepsilon(m_2) \ mod \ n^2 = g^{m_1+m_2} \cdot (\gamma_1\gamma_2)^n \ mod \ n^2$.

C. Protocol Steps

The Louis protocol consists of two phases of which the second one is optional. The first phase lets Alice determine whether Bob is nearby. (Recall that Alice considers Bob nearby if $d < 0$.) If this is the case, the (optional) second phase lets Alice and Bob learn each other's locations. On the other hand, if Alice detects misbehaviour, she can also use the second phase to detect the cheater. A detailed description is as follows.

1) First Phase: In order to initiate the protocol, Alice determines her location co-ordinates (x, y), a desired radius r, and chooses a random salt s_A. She then sends these values to Bob in the following message.

1. A $\rightarrow B : \varepsilon_A(x^2 + y^2), \varepsilon_A(2x), \varepsilon_A(2y), r, H(x, y, s_A)$

The aim is to let Bob compute the encrypted value of $d+k$ without the knowledge of Alice's co-ordinates, (x, y). This is one of the instances where the Paillier cryptosystem has been efficiently utilized. The hash value acts as a commitment of (x, y).

Bob checks the value of r. If he thinks r is too large, he aborts the protocol. Otherwise, he determines his location co-

ordinates (u, v), selects a random value k and computes:

$$\varepsilon_A(d+k) = \frac{\varepsilon_A(x^2+y^2) \cdot \varepsilon_A(u^2+v^2) \cdot \varepsilon_A(k)}{\varepsilon_A(2x)^u \cdot \varepsilon_A(2y)^v \cdot \varepsilon_A(r)^2}$$

This random value k is necessary to hide the value of d from Alice. Otherwise, knowing d and k will allow to compute D. Bob encrypts k using Trent's public key in order to keep the value of k confidential from Alice. Bob also chooses a random salt s_B for another hash function that will be similarly used as a commitment on part of Bob by Alice, later in the protocol (message 6).

2. B $\rightarrow A$: $\varepsilon_A(d+k), \varepsilon_T(k), H(u, v, s_B), H(k)$

Alice decrypts $\varepsilon_A(d+k)$ and sends it to Trent along with her signatures.

3. A $\rightarrow T$: d+k, $\varepsilon_T(k), sig_A(d+k), sig_A(\varepsilon_T(k))$

Trent decrypts $\varepsilon_T(k)$ and verifies Alice's signatures. Next, he computes d (by subtracting from $d+k$ the value of k, that he decrypted). If $d < 0$, Trent sets ans = 'YES' else, he sets ans = 'NO' and send it to Alice.

4. T $\rightarrow A$: ans, $\text{sig}_T(ans, sig_A(d+k), sig_A(\varepsilon_T(k)))$

Note that since Trent does not know the value of r, he can not calculate the actual distance between Alice and Bob.

Alice verifies Trent's signature and, if she sees ans is 'YES', she believes that Bob is nearby. Alice may initiate the second phase of the protocol if she is convinced with Trent's answer that Bob is nearby (and hence does not get suspicious). Alice may also initiate the second phase in order to find out the cheater when she gets suspicious on noticing an inconsistency between Trent's answer and her 'visual spotting'. The protocol is terminated if she is convinced with Trent's answer that Bob is not nearby or if only the first phase of the protocol is run.

2) Second Phase (Optional): At the end of the first phase, Alice learns Trent's answer to the question whether Bob is nearby. Since either Bob or Trent can cheat, she can not be sure of the answer. Such misbehaviour can be detected by Alice through visual spotting; then, to detect who is cheating Alice can execute the second phase. The term 'visual spotting' coined by Zhong et al. refers to any external means, through which Alice(or Bob) can detect inconsistencies. For example, if Alice receives 'NO'at the end of first phase, but 'spots' Bob nearby, she knows that cheating has taken place. Alice also executes the second phase, if she does not detect any misbehaviour and wants to find out the location coordinates of Bob when the answer says that they are nearby. The second phase is described below.

Alice sends ans to Bob along with signed copies of the messages she received from Trent in message 4.

5. A $\rightarrow B$: ans, d+k, $\text{sig}_A(d+k), sig_A(\varepsilon_T(k))$, $sig_T(ans, sig_A(d+k), sig_A(\varepsilon_T(k))), x, y, s_A$

Bob verifies all signatures. He then computes $H(x, y, s_A)$ and compares the hash value with the one provided by Alice in message 1. He also uses (x, y) to compute $d+k$ and compares it to the value received. If the values do not match, Bob aborts the protocol. Otherwise Bob reveals his location co-ordinates to Alice. He computes the $d+k$ value again in order to be assured that Alice didn't tamper with the $d+k$ value she received from Trent in message 2. He then sends his actual co-ordinates and the secret salt (s_B) so that Alice can also perform similar verifications.

6. B $\rightarrow A$: u, v, s_B, k

Alice computes $H(u, v, s_B)$ and $H(k)$ and compares the values with the hash values provided by Bob in message 2. Alice also computes $d+k$ based on (x, y), (u, v), and k and verifies whether it equals the decrypted value of $\varepsilon_A(d+k)$. If all the quantities are consistent, the protocol ends successfully. At the end of the protocol, once both the phases have been successfully run, both the parties (Alice and Bob) know each other's location co-ordinates.

IV. The Flaw

Any of the three principals Alice, Bob or Trent can misbehave during the protocol run. Alice or Bob may reveal locations, at which they are not present and Trent can misbehave by manipulating the answer it sends to Alice. The Louis protocol claims to directly detect such scenarios where one of the principals misbehaves but does not claim to detect the situations when Alice or Bob collude with Trent.

The detection of suspicious behaviour by Alice or Bob is through means external to the protocol, that is, the protocol does not provide means to detect misbehaviour, it only provides the means to detect as to who the misbehaving party is, if a suspicion arises. For example, if Alice visually spots Bob in the nearby vicinity after being told by Trent that Bob is not nearby, she gets suspicious and can now, according to the original protocol, detect who the misbehaving party is. The protocol also makes a hidden assumption, that has not been explicitly stated, which is that all the participants of the protocol have to remain static during the run, that is, movements of the participants are not accounted for. This is because if Bob moves to a different location during the run of the protocol, then an inconsistency will occur when Alice verifies them on receiving message 6 even if all the principals have been honest throughout the run. Also the outcome of Alice's visually spotting mechanism may be affected if Bob

changes his location during the run of the protocol.

Now let us see how the protocol attempts to detect the misbehaving parties, compromising one of the goals it was designed to achieve, in the process.

The visual spotting mechanism or the external means for Bob have been assumed in the analysis to support the statement in the original paper that if the second phase of the protocol is run and Bob detects suspicious behaviour, Bob uses mechanisms similar to Alice's to discover misbehaviour[1]. This mechanism is similar to Alice's (which has been described in the original paper), specifically if Bob does not spot Alice in the nearby area, he concludes she must be outside it.

We consider the following types of cheating.

- *Cheating by Trent* implies that he flips the answer after he calculates the distance. In other words, if the distance calculation suggests that Bob and Alice are nearby, Trent flips the answer and sends the response denoting that Bob and Alice are not nearby and vice versa.
- *Cheating by Bob or Alice* implies that they use fake co-ordinates, right from the start of the protocol run, instead of their actual location co-ordinates.

The protocol does not safeguard the privacy of a party which cheats. It should also be noted that if a participant tries to misbehave during the protocol, for instance, it changes its co-ordinates sent in the second phase, from those in first, it will be detected using the commitments made earlier. This kind of misbehaviour is hence detected.

We will now consider three cases for analysis, which are exhaustive under the assumptions being made. Note that unless the second phase is initiated by Alice, Bob will not detect any misbehaviour; it is only after that, that Bob needs to detect misbehaviour.

- Case 1: When only Trent Cheats
  In this case, if Trent sends the (flipped) answer as 'NO' (even when Alice and Bob are *actually* nearby), then Alice asks Bob to reveal his co-ordinates if she gets suspicious (that is to say that she spots Bob in her vicinity even when Trent said otherwise). So there is no problem in this situation as the parties exchanging the co-ordinates are actually nearby.
  On the other hand, if Trent sends the (flipped) answer as 'YES' (even when Alice and Bob are not *actually* nearby), then Alice reveals her co-ordinates (in order to initiate the second phase of the protocol) if she gets suspicious (that is to say that she fails to spot Bob in her vicinity even when Trent said otherwise). So here, there is a problem as Bob is forced to reveal his location coordinates to Alice even though he is not nearby. Also, even though Bob comes to know about Trent's cheating (as he can easily recalculate all quantities and yet fail to spot Alice in his vicinity), he has no way to convey this information to Alice without revealing his coordinates in the existing protocol. Also, Alice's coordinates are revealed to a party that is not nearby. If Bob aborts the protocol at this stage, he is labelled a cheater by Alice which may affect their friendship. The problem arises in this case as even when Bob has the capability to visually spot Alice, he can not convey this information to Alice as the protocol does not provide any channel to Bob to convey to Alice that he can also verify her absence (or presence) in his vicinity.

- Case 2: When only Bob Cheats
  In this case Trent is honest and performs the calculation according to the protocol. Now, when Bob cheats, and Alice gets suspicious, then in order to find out who the cheater is, Alice has to initiate the second phase of the protocol. This results in the disclosure of her location coordinates to a cheating party (Bob, who may not be nearby) and so Alice's privacy is compromised.

- Case 3: When only Alice Cheats
  Bob recalculates all the quantities and hence knows that Trent has not cheated. Now, since Bob has the 'visual spotting' capability he can conclude that Alice is the one who is cheating, on finding inconsistencies in Trent's answer and his 'visual spotting'. Hence, here he knows about Alice's cheating and will abort the protocol. This is the only case, where protocol works and where Bob's visual spotting mechanism will prevent his privacy from being compromised.

  We have shown that in two of the cases (cases 1 and 2), when a party cheats, the Louis protocol is unable to safeguard the location privacy of its users. In the next two sections we propose two fixes to solve the problem. The proposed fixes address the issue of flawless functioning of the Louis protocol under the original assumptions. They do not handle the situation when collusion between two parties takes place and assumes the parties to be static during the run of the protocol.

  When Alice gets suspicious of a supposed misbehaviour by either Trent or Bob, she initiates the second phase in order to detect the cheater, and if she does not receive Bob's cooperation, she deduces that he is the misbehaving party. The flaw, in the Louis protocol, is the result of Alice's faulty conclusion. This is caused as Alice does not have the information about Trent's behaviour (who could be cheating) when she gets the fifth message, when Bob gets to know about it. Secondly, on getting suspicious Alice has no choice but to execute the second phase to identify the cheater, resulting in revealing her coordinates unconditionally.

We propose two fixes to solve the problem. The first fix uses an additonal phase which effectively solves the problem of finding out the cheating party. But the fix only tones down the problem of location coordinates being revealed, as it requires the distance between Alice and Bob to be revealed when cheating takes place, in order to find the cheater. The second fix, which uses a probabilistic cut and choose method does not have this problem, but here the trade-off is in terms of cost efficiency.

V. A Fix Using An Additional Phase (First Fix)

In the original protocol, the second optional phase has two motives - detection of the cheater (when Alice gets suspicious) and exchange of the exact location coordinates. In our fix we aim to decouple these two motives by finding out the cheating principal in one phase (in case of Alice getting suspicious) and exchanging actual coordinates in another. The second optional phase of the Louis protocol above remains same in application, but its purpose is now limited only to the exchange of coordinates, when no suspicion arises. (Remember that cheating by Alice/Bob meant using fake coordinates right from the start; the case when they misbehave during the protocol is still handled by this phase itself.) To find out the cheating principal when Alice gets suspicious, we introduce another optional extra phase after the first phase, which requires Bob to let Alice know his actual relative distance from her and thus validate the correctness of Trent's reply sent to Alice. This decoupling allows Bob to remain free of the dilemma of whether to send the coordinates or not, and therefore the condition to Alice's faulty conclusion is avoided as well. The noteworthy fact here is that only one of the two optional phases in the fixed protocol can be executed in a protocol run since the additional phase is executed only when someone has cheated and as a result of that Alice gets suspicious. Once a cheating has taken place, there is no basis of exchanging the coordinates and hence the other phase can not be executed. Now, let us look at the proposed additional phase.

5'. A $\rightarrow B$: ans, d+k, sig$_A(d+k), sig_A(\varepsilon_T(k))$, $sig_T(ans, sig_A(d+k), sig_A(\varepsilon_T(k)))$

After the first phase is executed and Alice gets suspicious, it executes the additional optional phase to detect the cheater. Here, Alice sends to Bob the ans and $d+k$ values and the related signatures which she receives from Trent in message 4 but refrains from revealing the actual location coordinates (x, y) at this stage. When Bob receives $d+k$, he subtracts k from it and can thus check whether the answer Trent had sent to Alice is consistent, that is, if $d+k-k$ comes out to be less than zero then ans must equal 'YES' implying Bob was nearby and accordingly for the other case when it comes out to be greater than zero.

In case Bob detects an inconsistency, he will send the k value to Alice (Bob can not modify the k value due to the commitment he has sent previously), thus helping her realize that Trent was cheating, after which Alice can choose to abort the protocol at this stage itself. The security analysis presented in the next sub-section discusses other cases (of cheating by Alice or Bob himself) in depth.

6'. B $\rightarrow A$: k

A. Security Analysis

- Case 1: When only Trent cheats.
  In this case, when Alice gets suspicious, she sends message 5'; Bob in turn checks its consistency with Trent's answer. When Bob finds Trent's ans to be inconsistent with the value Alice has sent, he knows that Trent is cheating here and conveys this to Alice, by sending message 6'. Thus, the problem of finding out for sure who the misbehaving party is, is solved. The problem of Alice having to reveal her coordinates in order to find out the misbehaving party, is toned down in the sense that Alice simply has to reveal the distance to Bob.

- Case 2: When only Bob cheats.
  In this case, again, Alice sends message 5', after getting suspicious. Now, Bob's cheating will be caught if he replies to this message by sending message 6', because Alice will verify the consistency of Trent's ans, and hence will know that it is Bob who has cheated. Also, if Bob chooses not to reply in this situation, then again Alice will know that Bob is the cheater. The problem of Alice's coordinates getting revealed, is again toned down, just like in Case 1.

  Note that here Alice does not need to trust any one, to find out the cheater, once she is sure that a cheating has taken place. If Alice receives a k, after 5' from Bob, then she will first check using previous commitments if it is the same value sent earlier; if it is the same value, then she uses it to find d. If this value of d is consistent with Trent's previous ans, then it must be Bob who has cheated. If not, then Trent is the cheater. If Alice does not receive any reply for message 5', then she concludes that Bob is the cheater as he has not provided evidence to the contrary.

- Case 3: When only Alice cheats.
  Here, suppose Alice wants to use the new phase

for her advantage, to know Bob's distance attributes without any suspicion. Alice sends message 5', revealing her distance (which might be from some fake coordinates); Bob checks its consistency with Trent's ans, which agrees with the d value. Thus Bob knows that Trent has not cheated, and since Bob himself is honest, this means Alice's suspicion is unfounded, implying that Alice is misbehaving to know Bob's distance attributes. Thus Bob will not send message 6', and will sever his friendship with Alice. This is the reason, why Alice should be sure of the fact that someone has cheated, before sending message 5', because her only purpose is to know who the cheater is, once it has been confirmed that someone has cheated, through the visual spotting mechanism.
Also, note that the visual spotting mechanism for Bob is not required in this additional phase.

Using this fix, Alice and Bob are able to hide their actual coordinates when they are not nearby but this is done at the cost of revealing the relative distance between the two. Another fix that we propose in the next section overcomes this shortcoming as well.

VI. A Fix Using Cut and Choose Protocol (Second Fix)

The problem with the protocol, as stated above is that Alice does not have the information about Trent's behaviour when she gets fourth message, where Bob gets to know about it. This leads to Alice's incorrect conclusion that Bob is dishonest if his reply does not come. To fix this, we propose the following solution, which ensures Trent's honesty and ensures that Alice is sure upto a certain probability about the veracity of his reply before initiating the optional phase of the (original) protocol.

A cut and choose protocol is the one where one person divides goods or resource into what they believe are equal halves, and the other person chooses the half they prefer. Our fix is based on this concept, in the way that the initiating party, Alice, at the start of protocol, has N inputs $X_1, \cdots, X_N$ out of which Alice may be interested in maintaining the secrecy of only one input X_i, $1 \leq i \leq N$. She then sends all these N inputs to the other party, Trent who performs his part of the operation on all the N inputs without getting to know the secret input X_i as he does not know i. He then sends back all the N processed inputs to Alice who retrieves X_i out of $X_1, \cdots, X_N$. Trent can of course try and guess the value of i, the probability of which is $1/N$, but this can be controlled by Alice (by varying the value of N) according to the implementation needs.

We propose to use this idea in order to act as a fix. This needs tweaking of message 3 and 4 of the original protocol:

3. A $\rightarrow T$: d+k, $\varepsilon_T(k), sig_A(d+k), sig_A(\varepsilon_T(k))$

4. T $\rightarrow A$: ans, $\text{sig}_T(ans, sig_A(d+k), sig_A(\varepsilon_T(k)))$

Let $X_j = \{d_j + k_j, \varepsilon_T(k_j)\}$. Going by the above idea, Alice sends N such message sets, $X_1, \cdots, X_N$, to Trent, while the actual message (including the calculated $d+k$ value) she has received from Bob is X_i, $1 \leq i \leq N$, such that i is known only to Alice. The other dummy values are chosen so that Alice knows whether their 'd' value is greater than or less than zero. Trent then performs the computations on all these N message sets, thereby generating N response message sets, which he sends back to her as message 4. The probability of cheating by Trent, without getting detected is hence $1/N$. So in message 4 of the 'fixed' protocol, Alice now receives, $\{ans_1\}, \cdots, \{ans_N\}$.

3. A $\rightarrow T$: $\{\text{d}_1 + k_1, \varepsilon_T(k_1)\}, \cdots, \{d_N + k_N, \varepsilon_T(k_N)\}$
3'. T $\rightarrow A$: $\{\text{ans}_1, \cdots, ans_N\}$

Alice can then check the messages for tampering, that is, if she finds a different answer (other than what she expects, as she knows the 'd' value already) for any of the 'd' value other than that of X_i, she can be assured that Trent has resorted to cheating. If all the values are as expected then if any cheating takes place, Bob is held responsible. Alice sends the ans she received from Trent to get his signatures on it so that it can then be sent to Bob, like in the original protocol.

4. A $\rightarrow T$: $\{\text{ans}_i, d_i + k_i, \varepsilon_T(k_i), sig_A(d_i + k_i), sig_A(\varepsilon_T(k_i))\}$
4'. T $\rightarrow A$: $\text{sig}_T(ans_i, sig_A(d_i + k_i), sig_A(\varepsilon_T(k_i)))$

This modification allows to make the whole process more efficient than signing N different messages of which $N-1$ are dummy values but this is made possible at the cost of an additional communication step. The security of the fix depends on the magnitude of N and hence a trade-off between security and cost is required.

A. Security Analysis

- Case 1: When only Trent Cheats
  In this case, Alice knows about Trent's cheating immediately with a probability $(N-1)/N$.

- Case 2: When only Bob Cheats
  In this case, since Trent is not cheating (who can be caught, as mentioned in 1.1), Bob is blamed if any cheating occurs, so Alice can thus specifically

pinpoint the cheater.

– Case 3: When only Alice Cheats
  Bob recalculates all the quantities and hence knows that Trent has not cheated. Now, since Bob has the 'visual spotting' capability he can conclude that Alice is the one who is cheating, on finding that Trent's answer (ans is 'YES') is not consistent with his 'visual spotting' (when he does not spot Alice nearby). Hence, here he knows about Alice's cheating and will abort the protocol.

VII. Conclusion

In this paper we analyse the Louis protocol [1] and propose two fixes to a flaw in it. A direction of future work is to work on a solution that handles collusion between the parties. Additionally, as previously stated, the protocol with the first fix still requires some location attributes to be revealed to find out the cheater; and the second fix is probabilisitic, requring a trade off between security and cost. Hence, there still remains ample scope in the direction of finding a better solution to the problem.

A very plausible direction of work is the development of a practical implementation of the Louis protocol, along the lines of the $NearByFriend$ [27] application, which is an implementation of the Pierre protocol [1].

While the analysis done in this paper is a heuristic one, a more rigorous and formal analysis of such protocols could be a future line of research in this area.

References

[1] Ge Zhong, Ian Goldberg and Urs Hengartner, *Louis, Lester and Pierre: Three Protocols for Location Privacy*, Privacy Enhancing Technologies Symposium, 62-76, June 2007.

[2] MIT SENSEable City Lab, *iFind*. http://ifind.mit.edu. Accessed May 2008.

[3] P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, Advances in Cryptology-Eurocrypt '99, pages 223-238.

[4] R. Cheng, Y. Zhang, E. Bertino, S. Prabhakar, *Preserving User Location Privacy in Mobile Data Management Infrastructures*, 6th Workshop on Privacy Enhancing Technologies Proceedings, Lecture Notes in Computer Science 4258, Springer-Verlag, 393-412, June, 2006.

[5] B. Gedik, L. Liu, *Location Privacy in Mobile Systems: A Personalized Anonymization Model*, 25th International Conference on Distributed Computing Systems Proceedings, Lecture Notes in Computer Science 4258, 620-629, Springer-Verlag, June, 2005.

[6] M. Gruteser, D. Grunwald, *Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking*, First International Conference on Mobile Systems, Applications, and Services, MobiSys, May, 2003.

[7] M. F. Mokbel, C. Y. Chow, W. G. Aref, *The New Casper: Query Processing for Location Services without Compromising Privacy*, 32nd International Conference on Very Large Data Bases, 763-774, September, 2006.

[8] Keith B. Frikken, Mikhail J. Atallah, *Privacy Preserving Route Planning*, Proceedings of the 2004 ACM workshop on Privacy in the Electronic Society, 28, October, 2004, Washington DC, USA.

[9] S. D. Li, Y. Q. Dai, *Secure Two-Party Computational Geometry*, Journal of Computer Science and Technology, 20(2), 258-263, 2005.

[10] Y. L. Luo, L. S. Huang, H. Zhong, *Secure Two-Party Point-Circle Inclusion Problem*, Journal of Computer Science and Technology, 22(1), 88-91, 2007.

[11] Wenliang Du, Mikhail J. Atallah, *Secure Multi-Party Computation Problems and Their Applications: A Review and Open Problems*, Seventh International Workshop on Algorithms and Data Structures, 8-10, August, 2001, Providence, Rhode Island, USA.

[12] Wenliang Du, Mikhail J. Atallah, *Secure Multi-Party Computation Geometry*, Proceedings of the 2001 Workshop on New Security Paradigms, 10-13, September, 2001, Cloudcroft, New Mexico, USA.

[13] Wenliang Du, Mikhail J. Atallah,*Privacy-Preserving Cooperative Statistical Analysis*, Proceedings of the 17th Annual Computer Security Applications Conference, 102, 10-14, December,2001.

[14] C. Cachin, S. Micali, M. Stadler, *Computationally Private Information Retrieval with Polylogarithmic Communication*, Advances in cryptology: EUROCRYPT 1999, 308-318, 1998.

[15] G. M. Køien and V. A. Oleshchuk, *Location Privacy for Cellular Systems; Analysis and Solutions, Proceedings of 5th Workshop on Privacy Enhancing Technologies*, Lecture Notes in Computer Science 3856, Springer-Verlag, 40-58, May-June, 2005.

[16] K. Virrantaus, J. Markkula, A. Garmash, Y. V. Terziyan, *Developing GIS-Supported Location-Based Services*, Proceedings of First International Workshop on Web Geographical Information Systems, 423-432, Kyoto, Japan.

[17] T. Thomas, *Secure Two-party Protocols for Point Inclusion Problem*, CoRR, abs/0705.4185, 2007, http://arxiv.org/abs/0705.4185, DBLP, http://dblp.uni-trier.de.

[18] C. Cachin, *Efficient Private Bidding and Auctions with an Oblivious Third Party*, Proceedings of 6th ACM Conference on Computer and Communications Security, 120-127, November, 1999.

[19] Y. Lindell, *On the Composition of Secure Multi-party Protocols. Ph.D. Thesis. Department of Computer Science and Applied Mathematics. The Weizmann Institute of Science, Israel*, http://www.research.ibm.com/people/1/, Accessed July 2008.

[20] Bright Kite, *http://brightkite.com/*, Accessed July 2008.

[21] Dodge Ball, *http://www.dodgeball.com/*, Accessed July 2008.

[22] iPling, *http://www.ipling.com/english.html*, Accessed July 2008.

[23] Loki, *http://www.loki.com/*, Accessed July 2008.

[24] Loopt, *http://www.loopt.com/*, Accessed July 2008.

[25] Mobiluck, *http://www.mobiluck.com/*, Accessed July 2008.

[26] Same Cell, *http://www.samecell.com/*, Accessed July 2008.

[27] NearByFriend, *http://crysp.uwaterloo.ca/software/nearbyfriend/*, Accessed November 2008.