APPLAUS: A Privacy-Preserving Location Proof Updating System for Location-based Services

Zhichao Zhu and Guohong Cao
Department of Computer Science and Engineering
The Pennsylvania State University, University Park, PA 16802
{zzhu,gcao}@cse.psu.edu

***Abstract*—Today's location-sensitive service relies on user's mobile device to determine its location and send the location to the application. This approach allows the user to cheat by having his device transmit a fake location, which might enable the user to access a restricted resource erroneously or provide bogus alibis. To address this issue, we propose A Privacy-Preserving LocAtion proof Updating System (APPLAUS) in which co-located Bluetooth enabled mobile devices mutually generate location proofs, and update to a location proof server. Periodically changed pseudonyms are used by the mobile devices to protect source location privacy from each other, and from the untrusted location proof server. We also develop 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 exchange request based on their location privacy levels. APPLAUS can be implemented with the existing network infrastructure and the current mobile devices, and can be easily deployed in Bluetooth enabled mobile devices with little computation or power cost. Extensive experimental results show that our scheme, besides providing location proofs effectively, can significantly preserve the source location privacy.**

I. Introduction

Mobile devices, such as smartphones and PDAs, are playing an increasingly important role in people's lives. Location-based services take advantage of user location information and provide mobile users with a unique style of resource and services. Nowadays more and more location-based applications and services require users to prove their locations at a particular time. For example, "Google Latitude" and "Loopt" are two services that enable a user to track his friend's location in real-time. As location proof plays a critical role in enabling these applications, they are location-sensitive. The common theme across all these applications is that they offer a reward or benefit to users located in a certain geographical location. Thus, users have the incentive to lie about their locations.

There are many kinds of location-sensitive applications. One category is location-based access control. For example, a hospital might limit access to patient information by doctors or nurses unless they can prove that they are in a particular room of the hospital [16]. Meanwhile, one class of popular location-aware applications works only when users are able to prove their history locations [22], such as *auto insurance quote* in which auto insurance company might provide discounts to drivers who can prove that they take high-safety routes during their daily commutes, *fraud reduction on eBay* in which location proofs from the seller can serve as additional evidence that the seller's account has not been compromised by an attacker, *police investigations* in which police forces are interested in finding ways for people to be able to provide efficient and trusted alibis, 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.

All these location-sensitive applications require users to prove that they really are (or were) at the claimed location. Although most mobile users have devices capable of discovering their locations, they lack a mechanism to prove their current or past locations to applications and services. One possible solution is to build a trusted computing module on each mobile device to make sure trusted GPS data is generated and transmitted, but its cost will be very high. 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 can not be verified. Several systems have recently been designed to provide end users the ability to prove their locations through WiFi infrastructure. For example, [22] proposed a solution that is suitable for third-party attestation, but relies on a PKI and the wide deployment of 802.11 access-point infrastructure. [14] described a trusted computing platform that can be used to generate unforgeable geotags for mobile content such as photos and video, however, it relies on the expensive trusted computing module on mobile devices to generate proofs.

In this paper, we propose A Privacy-Preserving LocAtion proof Updating System (APPLAUS), which does not rely on the wide deployment of network infrastructure or the expensive trusted computing module. In APPLAUS, Bluetooth enabled mobile devices in range mutually generate location proofs, which are uploaded to a untrusted location proof server that can verify the trustworthy 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 changed pseudonyms at each mobile device to protect location privacy from each other, and from the untrusted location proof server. We use 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 based on their location privacy levels. Extensive experimental and simulation results show that our scheme, besides providing location proofs effectively, can significantly preserve the source location privacy.

The rest of the paper is organized as follows: we first introduce the preliminaries of our scheme in Section II. After that, Section III presents our location proof updating scheme. Section IV presents the source location privacy analysis and how to deal with colluding attacks. The performance of our scheme is evaluated in Section V. Finally, we describe the related work in Section VI and conclude the paper in Section VII.

II. Preliminaries

Our study can be applied to networks where mobile nodes are autonomous entities equipped with WiFi or Bluetooth enabled devices. For example, it could be a pervasive communication system (a mobile ad hoc network) such as a vehicular network [9], a delay tolerant network [5], or a network of directly communicating hand-held devices [1] in which mobile nodes in proximity automatically exchange information. In this paper, we focus on mobile networks where Bluetooth enabled devices such as cellular phones communicate with each other in short-range Bluetooth protocol. Given its appropriate range (about 10m) and low power consumption, Bluetooth is a natural choice for mutual encounters and location proof exchange.

A. Pseudonym

As commonly assumed in many networks, we consider an online Certification Authority (CA) run by independent trusted third party that pre-establishes the credentials for the devices. In line with many pseudonym approaches, to protect location privacy, we assume that prior to entering the network, every mobile node i registers with the CA that pre-loads a set of M public/private key pairs $K_i^{Pub}, K_i^{Prv}{}_{j=1}^{M}$. The M public keys K_i^{Pub} serve as the pseudonyms of node i. The private keys K_i^{Prv} enable node i to digitally sign messages, and the digital certificate validates the signature authenticity. When a node i receives a probe from another node, it controls the legitimacy of the sender by checking the certificate of the public key of the sender and the physical identity, e.g. Bluetooth MAC address. After that, i verifies the signature of the probe message. Subsequently, if confidentiality is required, a security association is established (e.g., with Diffie-Hellman).

B. Threat Model

We assume that an adversary aims to track the location of mobile nodes. An adversary can have the same credential as a mobile node and is equipped to eavesdrop communications. We assume that the adversary is internal, passive and global. By internal, we assume that the adversary is able to compromise or control individual mobile device and then communicate with others to explore private information, or individual devices would collude with each other to produce false proofs. By passive, we do not assume the adversary can perform active channel jamming, mobile worm attacks [28] or other denial-of-service attacks, since these attacks are not related to location privacy. By global, we assume that the adversary can monitor, eavesdrop and analyze all the traffic in its neighboring area due to short range communications.

In practice, the adversary can thus be a rogue individual, a set of malicious mobile nodes, or may even deploy its own infrastructure by placing eavesdropping devices in the network. In the worst case, the adversary obtains complete network coverage and tracks nodes throughout the entire network, e.g., it is possible that the untrusted location proof server might be compromised by the adversary and the location information can then be easily inferred by examining the records of location proofs. Therefore, we need to appropriately design and arrange the location proof records in the untrusted server so that no private information related to individual users would be revealed even after being compromised by adversary. Hence, the problem we tackle in this paper consists of collecting a set of location proofs for each peer node and protecting the location privacy of peer nodes from each other, from adversary, or even from the untrusted location proof server to prevent other parties from learning a node's past and current location.

C. Location Privacy Level

Consider a mobile network composed of N mobile nodes and each node has M pseudonyms. At time t, for each node i there are a group of $m(t)$ pseudonyms observed at the location proof server. Each pseudonym among the $m(t)$ pseudonyms can involve multiple location proofs across various locations $l_1, l_2, ..., l_n$ at different time $t_1, t_2, ..., t_n$. An adversary is able to correlate the location and time distribution of each pseudonym to see if two pseudonyms belong to the same node. For example, the adversary can observe a serial of location proofs with $m(T)$ pseudonyms during time T. He then compares the distribution of location proof set B of pseudonym b with the distribution of location proof set D of pseudonym d to determine if the two pseudonyms can be linked. Let $p_{d=b}$ = Pr(distribution D of pseudonym corresponds to distribution B of pseudonym b), the uncertainty of the adversary, and thus for our purposes the location privacy level of node i at time T, is

$$E_i(T) = -\sum_{d=1}^{m(T)} p_{d=b} log_2(p_{d=b}) \qquad (1)$$

The achievable location privacy depends on both the number of nodes $m(T)$ and the unpredictability of their whereabouts in the mix zone $p_{d|b}$. If a node i has only one pseudonym observed till time T, its identity is known to the adversary and its location privacy level is defined to be $E_i(T) = 0$. We can achieve the maximum entropy when every $p_{d=b}$ is close to 0, which means that the distribution of location proof sets for each pseudonym is undistinguishable.

III. The Location Proof Updating System

In this section we introduce the location proof updating architecture, the protocol, and how mobile nodes schedule their location proof updating to achieve statistically strong location privacy in APPLAUS.

A. Architecture

In APPLAUS, mobile nodes communicate with neighboring nodes through Bluetooth interface, and communicate with the untrusted server through cellular interface. Based on different roles they are playing in the process of location proof updating, they are categorized as Prover, Witness, Location Proof Server, Certificate Authority or Verifier. The architecture and message flow of APPLAUS is shown in Figure 1.

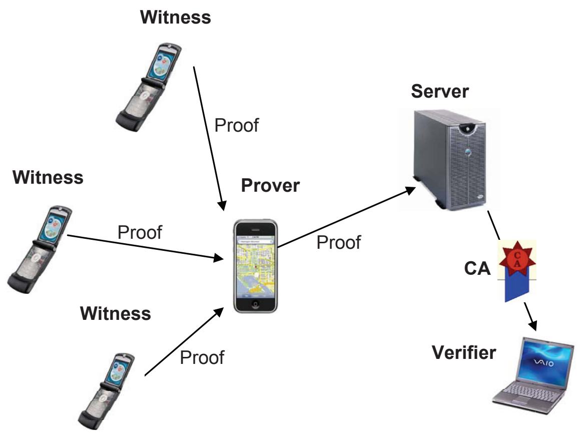


Fig. 1. Location Proof Updating Architecture and Message Flow

- **Prover**: Prover is the node who needs to collect location proofs from its neighboring nodes. When a location proof is needed at time t, the prover will broadcast a location proof request to its neighboring nodes through Bluetooth communication. If no positive response is received, the prover will generate a dummy location proof and submit it to the location proof server.
- **Witness**: Once a neighboring node agrees to provide location proof for the prover, this node becomes a witness of the prover. The witness node will generate a location proof and send it back to the prover.
- **Location Proof Server**: As our goal is not only to monitor real-time locations, but also to retrieve history location proof information when needed, a location proof server is necessary for storing the history records of the location proofs. It communicates directly with prover nodes who submit their location proofs. As the source identities of the location proofs are stored as pseudonyms, the location proof server is untrusted in the sense that even though it is compromised and monitored by attackers, it is impossible for the attacker to reveal the real source of the location proof.
- **Certificate Authority:** As commonly assumed in many networks, we consider an online Certification Authority (CA) run by an independent trusted third party. Every mobile node registers with the CA and pre-loads a set of public/private key pairs prior to entering the network. CA is the only party who knows the mapping between real identity and pseudonyms (public keys), and works as a bridge between the verifier and location proof server. It can retrieve location proof from the server and forward it to the verifier.
- **Verifier**: Verifier is a third-party user or an application who is authorized to verify a prover's location within a specific time period. The verifier usually has close relationship with the prover, e.x., friends or colleagues, to be trusted enough to gain authorization.

B. Protocol

When a prover needs to collect location proofs at time t, it executes the protocol in Figure 2 to obtain location proofs from the neighboring nodes within its Bluetooth communication range. Each node uses its M pseudonyms $P_{j=1}^{M}$ as its identity throughout the communication.

1) The prover broadcasts a location proof request to its neighboring nodes through Bluetooth interface according to its update scheduling. The request should contain the prover's current pseudonym P_{prov}, and a random number R_{prov}.
2) The witness decides whether to accept the location proof request according to its witness scheduling. Once agreed, it will generate a location proof for both prover and itself and send the proof back to the prover. This location proof includes the prover's pseudonym P_{prov}, prover's random number R_{prov}, witness's current timestamp T_{witt}, witness's pseudonym P_{witt}, and their shared location L. This proof is signed and hashed by the witness to make sure that no attacker or prover can modify the location proof and the witness cannot deny this proof. It is also encrypted by the server's public key to prevent from traffic monitoring or eavesdropping attacks.
3) After receiving the location proof, the prover is responsible for submitting this proof to the location proof server. It will also include its pseudonym P_{prov} and random number R_{prov} in the message.
4) An authorized verifier can query the CA for location proofs of a specific prover. This query contains a real identity and a time interval. The CA first authenticates the verifier, and then converts the real identity to its corresponding pseudonyms during that time period and retrieves their location proofs from the server.
5) The location proof server only returns hashed location rather than the real location to the CA, who then forwards to the verifier. The verifier compares the hashed location with the claimed location acquired from the prover to decide if the claimed location is authentic.

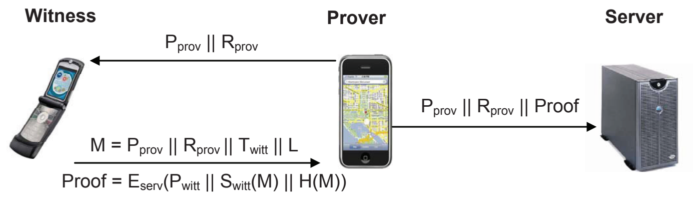

Witness
Prover
Server
$P_{prov} \| R_{prov}$
$P_{prov} \| R_{prov} \| Proof$
$M = P_{prov} \| R_{prov} \| T_{witt} \| L$
$Proof = E_{serv}(P_{witt} \| S_{witt}(M) \| H(M))$

Fig. 2. Location Proof Updating Protocol

In order to prevent the CA from knowing locations of a real identity, we have location proof server calculate the hash of each location and only send the hashed locations to the CA in step 5. In this way, the following property can be achieved.

Definition 1 (Separation of Privacy Knowledge): The knowledge of the privacy information is separately distributed to the location proof server, the CA, and the verifier, respectively. Each party only has partial knowledge.

The privacy property of our protocol is ensured by the separation of privacy knowledge: the location proof server only knows pseudonyms and locations, the CA only knows the mapping between the real identity and its pseudonyms, while the verifier only knows the real identity and its authorized locations. Attackers are unable to learn a user's location information without integrating all the knowledge. Therefore, compromising either party of our system does not reveal privacy.

C. Scheduling Location Proof Updates

As discussed before, the adversary might obtain complete coverage and track nodes throughout the entire network, by compromising the location proof server and obtain all history location proofs. Therefore, we need to appropriately design and arrange the location proof updating schedules for both prover and witness so that no source location information related to individual user would be revealed even when the server is compromised by adversary.

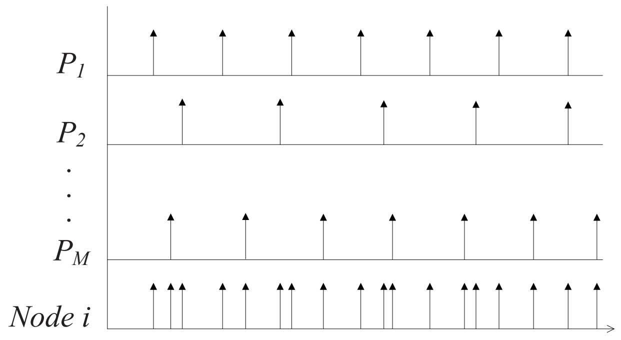


Fig. 3. Example of individual pseudonym update vs. entire node update

Suppose a mobile node i has a set of pseudonyms $P_1, P_2, \cdots, P_M$ which change periodically, and distinct parameters $\lambda_1, \lambda_2, \cdots, \lambda_M$ for each pseudonym are pre-determined. If each pseudonym P_j updates its location proofs (including dummy proofs) such that the inter-update interval follows Poisson distribution with parameter λ_j, as in Figure 3, then the entire inter-update intervals for node i follow Poisson distribution with a parameter of $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_M$. As will be discussed in the next section, it has the properties of pseudonym unlinkability and statistically strong source location unobservability. The detailed scheduling protocol for the prover is shown in Algorithm 1. The pre-defined updating parameter λ determines how frequently location proofs are updated.

On the other hand, the location privacy of witness nodes varies depending on time and location when they exchange

Algorithm 1 Location Proof Updating Scheduling for the prover

```
Input: updating parameter $\lambda$;
1: generate $M$ distinct parameter $\lambda_1, \lambda_2, \cdots, \lambda_M$ such that
   $\lambda_1 + \lambda_2 + \cdots + \lambda_M = \lambda$
2: for each pseudonym $i$ do
3:   while current timestamp $t$ follows Poisson distribution
     with $\lambda_i$ do
4:     send location proof request
5:     if request is accepted then
6:       submit location proof
7:     else
8:       generate and submit dummy proof
9:     end if
10:  end while
11: end for
```

location proofs for others. It is thus desirable to protect the location privacy in a user-centric manner, such that each user can decide when and where to protect its location privacy. User-centric location privacy [10] [11] [15] is a distributed approach where each mobile node locally monitors its location privacy level over time. The user centric approach is easily scalable and permits a more fine-grained approach to maintain location privacy. In our model, each mobile node monitors and measures its own privacy level in real-time and acts upon it by deciding whether and when to accept a location proof exchange request. When a location proof exchange request is received in proximity, it calculates the privacy loss between the next scheduled updating time and the current updating time. The privacy loss of node i is defined as follows:

$$\Delta = \frac{E_i(t') - E_i(t)}{E_i(t)} \tag{2}$$

where $E_i(t)$ is the location privacy level when the location proof exchange request is accepted while $E_i(t')$ is the location privacy level at the next scheduled location proof updating cycle. The difference between the two indicates the privacy loss if this location proof exchange request is accepted. The location proof exchange request is only accepted when the privacy loss is less than a predefined threshold. The drawback of the user-centric model is that nodes may have misaligned incentives (i.e., different privacy requirement), which can lead to failed attempts to achieve location proofs. We use dummy proofs in Algorithm 1 to deal with failed attempts. The detailed scheduling protocol for witness is presented in Algorithm 2.

IV. Security Analysis and Countermeasures

In this section, we discuss the security property in terms of source location privacy and colluding attacking, as well as the countermeasures for these threats.

A. Source Location Privacy

Now we look at how an adversary may reveal location information by analyzing the location proof history. Suppose the attacker has sufficient resources (e.g., in storage, computation and communication). First, the attacker may simply monitor

Algorithm 2 Scheduling Location Proof Updates at Witnesses

```
Input: time $t$ of incoming location proof exchange request;
1: calculate location privacy loss $\Delta$ when assuming the incoming request is accepted
2: if $\Delta > \epsilon$, $\epsilon$ is pre-defined location privacy loss threshold then
3:     deny location proof exchange request
4: else
5:     accept location proof exchange request
6: end if
```

and examine the content of a record that may contain the user's identity and location. Second, even if the user's ID is encrypted or pseudonymized, it is easy for the adversary to trace back all the location activities related to the same ID once its pseudonym is discovered. Third, even though the user's pseudonyms change periodically, it is still possible for the adversary to infer this user's other pseudonyms from one pseudonym if these pseudonyms change at similar time or locations. Moreover, the attacker may perform more advanced traffic analysis including rate monitoring and location correlation. In a rate monitoring attack, the attacker tries to monitor and correlate location proof updating rates from different pseudonyms. In a location correlation attack, the attacker may observe the correlation in updated location between a node and its neighbor, attempting to deduce a relationship.

According to [20], a mechanism to achieve anonymity appropriately combined with dummy traffic yields unobservability, which is the state that Items of Interests (IOIs) are indistinguishable from any IOI of the same type. All the subjects and events under consideration constitute an unobservability set. In our case, the unobservability set consists of all the M pseudonyms for each node in the network. Specifically, we are interested in the privacy property of source location unobservability. Therefore, we have the following definition on source location **unobservability**.

Definition 2: Source location unobservability is a privacy property that can be satisfied if an attacker cannot determine the real identity of mobile nodes through full observation of the location proof records. That is, for each possible observation O that an attacker can make, if the probability of an identity I is equal to the probability of I given O, that is: $\forall O, P(I) = P(I|O)$, then I is called unobservable.

Based on the above definition, we have the following definition on pseudonym unlinkability.

Definition 3: A system has the property of **pseudonym unlinkability** if any pseudonyms $P_1, P_2, \cdots, P_M$ of an identity I presented in the location proof records cannot be inferred from one to another: $\forall i, j, \forall O, prob(P_i|O) \neq prob(P_j|O), i, j \in \{1, 2, ..., M\}, i \neq j$.

Obviously, a system satisfies source location unobservability if and only if it has the property of pseudonym unlinkability. We need to design a probabilistic solution, which can provide the chance of reducing the latency as much as possible while still providing a satisfactory degree of event unobservability. Let the inter-update delay (d) between location proof updates $k (k > 0)$ and $k+1$ from a pseudonym $i (1 \leq i \leq M)$ of a mobile node be $d_k^i = t_{k+1}^i - t_k^i$, where t_k^i is the updating time of location proof k from pseudonym i. A global attacker can observe a sequence of inter-update delays, which can be represented as a distribution $X_i = d_1^i, d_2^i, \cdots$. In our case of **statistically strong source location unobservability**, distributions of inter-update delays by different pseudonyms are statistically distinct from each other. They are distinct from each other in the sense that by a statistic test one cannot correlate them with each other. We have the definition of statistically distinct distributions as follows.

Definition 4: Two probabilistic distinct distribution X_i and $X_j (1 \leq i, j \leq M, i \neq j)$ are statistically distinct from each other if they follow the same type of probabilistic distribution with distinct parameter.

Take the Poisson distribution as an example. This distribution has only one parameter λ. Hence, if two probabilistic distributions are both Poisson distributions with distinct means, they are statistically distinct from each other. Clearly, the more parameters a distribution has, the harder it is to prove its statistical distinction. In our scheme, we decide to use Poisson distribution to control the rate of location proof updating, not only due to its one-parameter advantage, which makes it relatively easy to achieve source location unobservability, but also because Poisson distribution is most similar to the property of contact rate of mobile nodes in an intermittent connected network. More specifically, we have the following Lemma.

Lemma 1: Suppose a mobile node has a set of pseudonyms $P_1, P_2, ..., P_M$ which change periodically. Each pseudonym i sends out its location proof updates (including dummy update) whose inter-update delay follows Poisson distribution with a parameter of λ_i. Then all the inter-update delays of this mobile node follow Poisson distribution with a parameter of $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_M$.

Proof 1: Without losing generality, we assume the mobile node changes its pseudonyms in the order of $P_1, P_2, ..., P_M$. As each pseudonym i follows Poisson distribution $X_i = P(X = t) = \frac{\lambda_i^t e^{-\lambda_i}}{t!}$, the distribution of the combination of pseudonyms P_i and P_j is

$$\begin{aligned} Y &= P(X_i + X_j = k) && (3) \\ &= \sum_{t=0}^{k} P(X_i = t) P(X_j = k - t) && (4) \\ &= \sum_{t=0}^{k} \frac{\lambda_i^t e^{-\lambda_i}}{t!} \cdot \frac{\lambda_j^(k-t) e^{-\lambda_i}}{(k-t)!} && (5) \\ &= \frac{\lambda_i^t e^{-\lambda_i}}{t!} + \frac{\lambda_j^t e^{-\lambda_j}}{t!} && (6) \\ &= \frac{(\lambda_i + \lambda_j)^t e^{-(\lambda_i + \lambda_j)}}{t!} && (7) \end{aligned}$$

That is, the sum of two independent Poisson distributions with parameters of λ_i and λ_j also follow a Poisson distribution with parameter $\lambda_i + \lambda_j$. It is easy to extend that the sum of all pseudonyms also follows Poisson distribution with a parameter of $\lambda = \lambda_i + \lambda_2 + \cdots + \lambda_M \blacksquare$.

Therefore, if each mobile node in the network chooses M distinct parameters from λ_1 to λ_M for its M pseudonyms, and schedules location proof updating based on the aforementioned Poisson distributions, then the location proof records in the server have the properties of pseudonym unlinkability and statistically strong source location unobservability.

B. *Colluding Attacks*

Another threat exists when two nodes collude with each other to generate bogus location proofs. When a malicious node C_1 needs to prove himself in New York City, he can have another colluding node C_2 to mutually generate bogus location proofs for him, with location tag of New York City. Generally, such attacks can be identified by using threshold based solution or by looking into the location traces. In threshold based solution, the system can require the prover to obtain a threshold number of witness nodes, and hence can deal with some colluding attacks. However, since it is hard for the prover to always find enough number of witness nodes, we also use the following solution. The server has information about the numbers of pseudonyms at particular time and location. This information can be used to estimate whether a prover lies about not finding enough peers or always finding the same peer based on some statistical techniques.

More specifically, we are considering two-level cross-validation from both location proof server and CA point of view and then integrating the results to ensure accuracy. The server-level validation is performed on individual location proof based on its timestamp and location information, where all concurrent and co-located location proofs from other pseudonyms are used to verify the reliability of the target location proof. For example, when a pair of location proofs by pseudonyms P_A and P_B are uploaded, the server checks if there are concurrent and co-located location proofs from other pseudonyms, which do not have any interaction with P_A and P_B. If there are such location proofs, P_A and P_B are suspicious to be colluders, and we then assign an appropriate trust level for the location proof. Future research issues lie in how to design trust level metric appropriately for each location proof, and evaluate an optimal server-level threshold to rule out the bogus location proofs.

As the CA has the knowledge of mapping between pseudonyms and real identity, as well as the trust levels of all individual location proofs for a real identity which are received from the location proof server, it is able to identify the trust level of a node in continuous time serial. The CA can calculate the average and variance of trust levels from individual location proofs, or the ratio of legitimate proofs over all the proofs, to determine the trust level of a node.

V. Performance Evaluations

In this section, we consider deployment feasibility for APPLAUS, including the computation and storage constraint, power consumption, as well as the proof exchange latency. We also use simulations to compare the performance of APPLAUS with a baseline scheme, and evaluate the privacy level against powerful statistical analysis attacks.

A. *Prototype Implementation*

To study the feasibility of our scheme, we have implemented an experimental prototype of APPLAUS based on the technique presented in the previous sections. The prototype has two software components: client and server. The client is implemented in JAVA on top of Android Developer Phone 2 (ADP2), which is equipped with 528MHz chipset, 512MB ROM, 192MB RAM, Bluetooth and GPS module, and running Google Android 1.6 OS. This device can communicate with the server anytime through AT&T's 3G wireless data service. The server component is implemented in C++ on a T4300 2.1GHz 3GB RAM laptop. It stores the uploaded historical location proof records and manages corresponding indices using MySQL 5.0. We use two android phones to communicate with each other to test the practicality of our scheme.

Our client code consumes only 80KB of data memory. When running, less than 2.5% of the available memory is taken. We measure the CPU utilization of our client code using a monitoring application, which allows one to monitor the CPU usage of all the processes running on the phone. When our application is in standby, the CPU utilization is about 0.5%, indicating that listening to incoming Bluetooth inquiries requires very low computation. The CPU utilization goes to 3% and 5% respectively when communicating with another device and with the server, due to different communication interfaces. We observe that the CPU utilization reaches the highest level of 10% when a location proof packet is generated, in which heavy computations such as authentication and encryption/decryption are involved.

We also measure the performance with two metrics: proof exchange time latency and power consumption. Figure 4 shows the latency to complete a location proof exchange process for different sizes of public/private key pair. The key size determines the number of bits in the encrypting keys as well as the size of the pseudonyms. Larger key size can provide better security level, but involves more computation and storage resources. It can be seen from the figure that 128-bit key is the best choice since it can provide adequate security level, without introducing too much delay. The distance between the two devices when they exchange location proof also has some effect on the latency, where longer distance means longer delay due to the transmission signal strength. As has been established in earlier studies [13], more than 80% of contact durations are less than 10 seconds, and thus there is no problem for our proof exchange process to be finished within the contact duration.

Figure 5 measures the power consumption under different Bluetooth status. There are three status: inquiry, standby and proof exchange. The inquiry status is used to discover other Bluetooth devices which are within communication range, and also send out proof requests. The inquiry process continues for a pre-specified time, until a pre-specified number of units

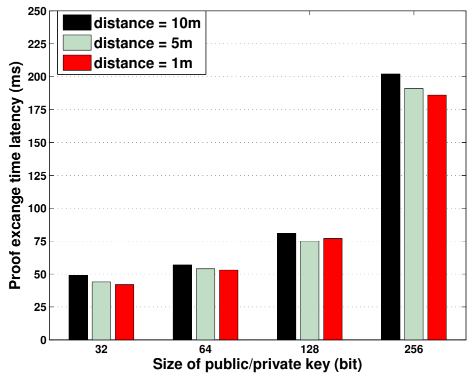


Fig. 4. The latency to perform a location proof exchange under different size of public/private key pair

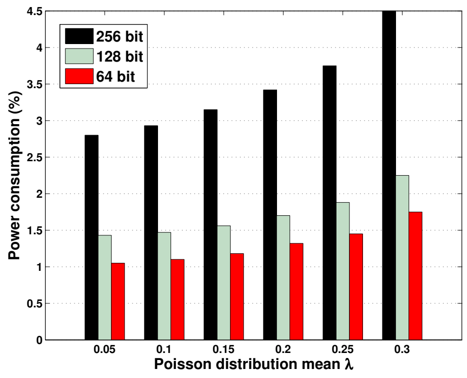


Fig. 6. Percentage of power consumption under different Poisson distribution mean λ

have been discovered or until it is stopped explicitly. Bluetooth devices that only listen to inquiry messages are in standby status. In our system, inquiry and standby are mutual exclusive at any time. The device enters the proof exchange status when it exchanges location proofs with others. The most frequent status is standby, which consumes only less than 0.1mW of power with any communication distance. The proof exchange status consumes the most amount of power and deteriorates with increasing communication distance; however, it will not appear until the next location proof updating cycle.

It is also helpful to show how the power consumption affects the daily normal usage of the mobile device. Figure 6 shows the percentage of power consumption on location proof exchanges by Poisson parameter λ. As the Poisson distribution mean λ becomes larger, the power consumption percentage increases, due to the more frequent location proof updating activities. It is also confirmed that 256-bit key consumes much more energy than 128-bit key, which is proved to be the most appropriate size for public/private key pair. It is easy to see our scheme does not increase the power consumption too much (with less than 2.5% power consumption).

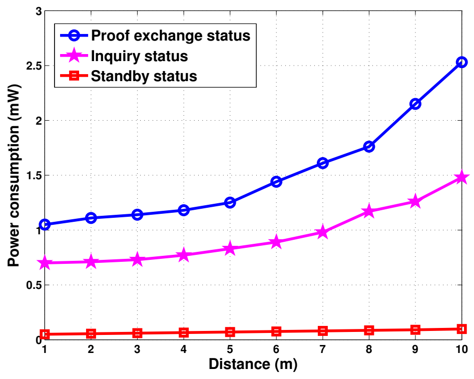


Fig. 5. Power consumption under different Bluetooth status and different communication distance

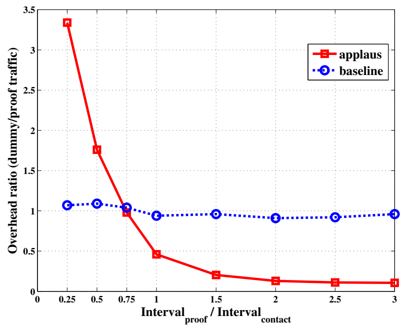
B. Simulation Results

In our simulations, 1000 mobile nodes are deployed in a 3km $\times$ 3km area. For each node, the range of Bluetooth transmission is 10m. We use the Levy walk mobility model [21] [6] to generate synthetic device contact events. Previous work has shown that the Levy walk model can describe the mobility patterns of a human being relatively well for a campus-sized area. A contact between two nodes happens when the nodes are within 10m range of each other.

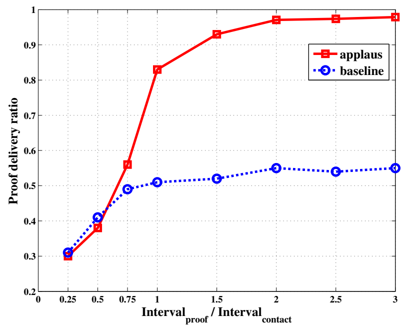
For each simulation run, we generate a Levy trace with a certain contact rate and initiate the location proof updating process with various time intervals. We repeat this experiment 100 times, choosing different contact rates on each run. Each node has $M = 10$ pairs of 128-bit public/private keys and a Poisson distribution parameter λ, which is used to determine when to change pseudonyms. λ is divided into 10 different numbers: $\lambda_1, \lambda_2, \cdots, \lambda_{10}$, where $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_{10}$. We define a parameter δ, which is the standard deviation of two consecutive λ_i and λ_{i+1} (assuming λ_i and λ_{i+1} are in non-decreasing order), to control how to cut λ. Another parameter ϵ is chosen by each node as a threshold to determine whether to accept a location proof exchange request based on the user-centric privacy level.

During our evaluation, we use three metrics: message overhead ratio, proof delivery ratio, and average delay. The message overhead ratio is defined as the ratio of dummy traffic and real proof traffic. The proof delivery ratio is the percentage of location proof message that is successfully uploaded to the location proof server. The average delay is the time difference between the time when a location proof update is needed and when the location proof message has reached the location proof server. We compare our APPLAUS scheme with a baseline scheme in terms of all metrics. In the baseline scheme, each node does not alter pseudonyms based on Poisson distribution. Rather, it uses a constant rate to upload location proofs. Unlike APPLAUS where two nodes mutually exchange location proofs, the baseline scheme only uploads its own location proof if there is a proof available. A dummy message is uploaded instead when there is no proof available.

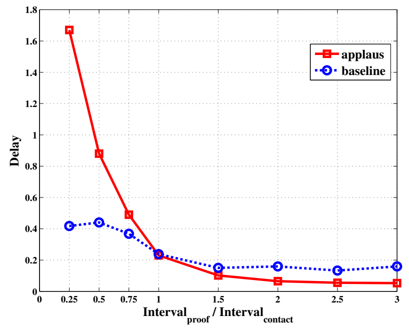
Figure 7 shows the performance comparison between our scheme and the baseline scheme under different ratio of $Interval_{proof}$ and $Interval_{contact}$. Here, $Interval_{proof}$ is the required interval between two location proof updates, while $Interval_{contact}$ is the mean real contact interval. Let



(a) Overhead Ratio



(b) Proof Delivery Ratio



(c) Average Delay

Fig. 7. Performance under different ratio of $Interval_{proof}$ and $Interval_{contact}$

$\omega = Interval_{proof}/Interval_{contact}$. Figure 7 (a) shows that APPLAUS outperforms baseline on overhead ratio when ω is larger than 0.75. When $\omega > 1.5$, the overhead ratio of APPLAUS decreases to as low as 0.2. The proof delivery ratio in Figure 7 (b) also reaches 93% when $\omega > 1.5$. In Figure 7 (c), APPLAUS and baseline have similar average delay when $\omega > 1$, in which the delay is measured as the unit of $Interval_{proof}$. When $\omega > 1.5$, the delay becomes lower than 0.15 of $Interval_{proof}$. We can conclude that when $\omega > 1.5$, that is, when the location proof update interval is at least 1.5 of the contact interval, the performance of APPLAUS reaches an adequate level. The performance is not improving significantly after $\omega = 2$. Therefore, an appropriate ratio ω between $Interval_{proof}$ and $Interval_{contact}$ should be carefully chosen between 1.5 and 2.

C. Privacy

As stated in the previous section, our location proof updating system has the property of pseudonym unlinkability and statistically strong source location unobservability; i.e., the source privacy of location information can be well preserved. In this subsection, we evaluate the robustness of APPLAUS in defending against powerful traffic analysis and statistical test.

To detect if two pseudonyms belong to the same source, the attacker can check whether two probabilistic distributions of proof message time intervals from the two pseudonyms are identical. For the attacker, the hypotheses of the test are:

- H_0 - the two pseudonyms belong to the same source.
- H_1 - the two pseudonyms belong to different source.

When the attacker makes a decision, there are some risks to get wrong decision. The decision is called a detection, if H_0 is accepted when it is actually true. If H_0 is in fact true, accepting H_1 is a false negative. On the other hand, if H_1 is in fact true, accepting H_0 is a false positive. False positive has no negative effect on privacy since taking two different pseudonyms as the same would not help identifying the real source. We focus on false negative which indicates the percentage of cases that has not been detected by the attackers.

To test false negative, we use the same simulation setup as previous section and repeatedly perform K-S test [19] on the distributions under different standard deviation δ (ranging from 0.2 to 1) and threshold ϵ (ranging from 0.02 to 0.1). Based on the test results, we get the false negative rate as shown in Figure 8. The x-axis is the threshold ϵ of each user. The smaller ϵ is, the less likely a location proof exchange request would be accepted. The y-axis represents the standard deviation δ of two Poisson distribution means. Obviously, larger δ indicates wider difference between the two Poisson distributions, which is harder for attackers to differentiate. We can observe from the figure that the false negative rate of the attacker's test is high (above 90%), especially when $\epsilon < 0.04$ and $\delta > 0.5$, which indicates that as long as the parameters of ϵ and δ are appropriate selected, the privacy of our system can be well preserved.

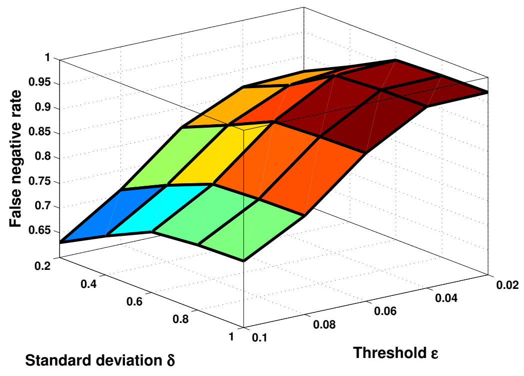


Fig. 8. False negative rate under different parameter of δ and ϵ

VI. Related Work

Recently, several systems have been proposed to give end users the ability to prove that they were in a particular place at a particular time. [23] [2] relies on the fact that nothing is faster than the speed of light in order to compute an upper-bound of a user's distance. [3] proposes challenge-response schemes, which use multiple receivers to accurately estimate a wireless node location using RF propagation characteristics. However, dedicated hardware is required to measure signal round trip time with very high precision and negligible processing delay. [22] proposes a solution that is suitable for third-party attestation, but relies on a PKI and wide deployment of 802.11 access-point infrastructure. APPLAUS uses a peer-to-peer approach and requires no change to the existing infrastructure. In [14], the authors describe a secure localization service that can be used to generate unforgeable geotags for mobile content such as photos and video. However this work relies on the high-cost trusted computing module. SmokeScreen [4] introduces a presence sharing mobile social service between co-located users which relies on centralized, trusted brokers to coordinate anonymous communication between strangers. SMILE [17] [18] allows users to establish missed connections and utilizes similar wireless techniques to prove when a physical encounter occurred. However, this service does not reveal the actual location information to the service provider thus can only provide location proofs between two users who have actually encountered. APPLAUS can provide location proofs to third-party by uploading real encounter location to the untrusted server while maintaining location anonymity.

Another focus of this paper is location privacy and source location anonymity. There has been many previous work in the area of location privacy for wireless users and devices. [8] proposes to reduce the accuracy of location information along spatial and/or temporal dimensions. This basic concept has been improved by a series of works [7] [12]. All the above techniques cloak a user's locations with its current neighbors by trusted central servers which is vulnerable to DoS attacks or to be compromised. Our approach does not require the location proof server to be trustworthy. Xu *et al* [25] [26] propose a feeling-based model for location privacy protection in location-based services, which allows a service user to express his privacy requirement. Identifying a fundamental tradeoff between performance and privacy, Shao *et al* [24] [27] propose a notion of statistically strong source anonymity in sensor networks for the first time. Our scheme uses similar source location unobservability concept in which the real location proof message is scheduled through statistical algorithms. However, their focus is to generate identical distributions between different nodes to hide the real event source, while our focus is to design distinct distributions between different pseudonyms to protect real identity.

VII. Conclusions

This paper proposed a privacy-preserving location proof updating system, called APPLAUS, in which co-located Bluetooth enabled mobile devices mutually generate location proofs, and upload to the location proof server. We use statistically changed pseudonyms for each device to protect source location privacy from each other, and from the untrusted location proof server. We also develop 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 exchange request based on their location privacy levels. To the best of our knowledge, this is the first work to address the joint problem of location proof and location privacy. Extensive experimental and simulation results show that our scheme can provide location proofs effectively while preserving the source location privacy at the same time.

VIII. Acknowledgements

This work was supported in part by Army Research Office under MURI grant W911NF-07-1-0318.

References

[1] Social Serendipity. *See http://http://reality.media.mit.edu/serendipity.php*.
[2] S. Brands and D. Chaum. Distance-bounding protocols. In *Advances in Cryptology-EUROCRYPT*, 1994.
[3] S. Capkun and J.-P. Hubaux. Secure positioning of wireless devices with application to sensor networks. In *IEEE INFOCOM*, 2005.
[4] L.P. Cox, A. Dalton, and V. Marupadi. Smokescreen: flexible privacy controls for presence-sharing. In *ACM MobiSys*, 2007.
[5] K. Fall. A delay-tolerant network architecture for challenged internets. In *ACM SIGCOMM*, 2003.
[6] W. Gao and G. Cao. Fine-grained mobility characterization: steady and transient state behaviors. In *Proceedings of the eleventh ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2010.
[7] B. Gedik and L. Liu. A customizable k-anonymity model for protecting location privacy. In *IEEE ICDCS*, 2005.
[8] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *ACM MobiSys*, 2003.
[9] H. Hartenstein and KP Laberteaux. A tutorial survey on vehicular ad hoc networks. *IEEE Communications Magazine*, 2008.
[10] 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. In *ACM MobiSys*, 2008.
[11] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Preserving privacy in GPS traces via density-aware path cloaking. In *ACM Conference on Computer and Communications Security (CCS)*, 2007.
[12] T. Jiang, H.J. Wang, and Y.-C. Hu. Location privacy in wireless networks. In *ACM MobiSys*, 2007.
[13] V. Kostakos. Experiences with urban deployment of Bluetooth (given at UCSD), Mar. 2007.
[14] V. Lenders, E. Koukoumidis, P. Zhang, and M. Martonosi. Location-based trust for mobile user-generated content: applications, challenges and implementations. In *ACM HotMobile*, 2008.
[15] M. Li, K. Sampigethaya, L. Huang, and R. Poovendran. Swing & swap: user-centric approaches towards maximizing location privacy. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*, 2006.
[16] W. Luo and U. Hengartner. Proving your location without giving up your privacy. In *ACM HotMobile*, 2010.
[17] J. Manweiler, R. Scudellari, Z. Cancio, and L.P. Cox. We saw each other on the subway: secure, anonymous proximity-based missed connections. In *ACM HotMobile*, 2009.
[18] J. Manweiler, R. Scudellari, and L.P. Cox. SMILE: Encounter-based trust for mobile social services. In *ACM CCS*, 2009.
[19] F.J. Massey Jr. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.
[20] A. Pfitzmann and M. Hansen. Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management–a Consolidated Proposal for Terminology (Version v0. 31 Feb. 15, 2008). *See http://dud.inf.tu-dresden.de/literatur*.
[21] I. Rhee, M. Shin, K. Lee, and S. Chong. On the Levy-walk Nature of Human Mobility. In *IEEE INFOCOM*, 2007.
[22] S. Saroiu and A. Wolman. Enabling new mobile applications with location proofs. In *ACM HotMobile*, 2009.
[23] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *ACM WiSE*, 2003.
[24] M. Shao, Y. Yang, S. Zhu, and G. Cao. Towards statistically strong source anonymity for sensor networks. In *IEEE INFOCOM*, 2008.
[25] T. Xu and Y. Cai. Exploring historical location data for anonymity preservation in location-based services. In *IEEE INFOCOM*, 2008.
[26] T. Xu and Y. Cai. Feeling-based location privacy protection for location-based services. In *ACM CCS*, 2009.
[27] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao. Towards event source unobservability with minimum network traffic in sensor networks. In *ACM WiSec*, 2008.
[28] Z. Zhu, G. Cao, S. Zhu, S. Ranjan, and A. Nucci. A social network based patching scheme for worm containment in cellular networks. *IEEE INFOCOM 2009*.