Who, When, and Where? Location Proof Assertion for Mobile Devices

Rasib Khan, Shams Zawoad, Md Munirul Haque, Ragib Hasan

'Who, When, and Where?' Location Proof Assertion for Mobile Devices

Rasib Khan, Shams Zawoad, Md Munirul Haque, and Ragib Hasan

Department of Computer and Information Sciences
University of Alabama at Birmingham
{rasib, zawoad, mhaque, ragib}@cis.uab.edu

Abstract. In recent years, location of mobile devices has become an important factor. Mobile device users can easily access various customized applications from the service providers based on the current physical location information. Nonetheless, it is a significant challenge in distributed architectures for users to prove their presence at a particular location in a privacy-protected and secured manner. So far, researchers have proposed multiple schemes to implement a secure location proof collection mechanism. However, such location proof schemes are subject to tampering and not resistant to collusion attacks. Additionally, the location authority providing a location proof is assumed to be honest at all times. In this paper, we present the fundamental requirements of any location proof generation scheme, and illustrate the potential attacks possible in such non-federated environments. Based on our observations, we introduce a concept of witness oriented endorsements, and describe a collusion-resistant protocol for asserted location proofs. We provide an exhaustive security analysis of the proposed architecture, based on all possible collusion models among the user, location authority, and witness. We also present a prototype implementation and extensive experimental results to adjust different threshold values and illustrate the feasibility of deploying the protocol in regular devices for practical use.

Key words: Location Assertion, Location Proof, Proof Protocol, Security, Witness Endorsement.

1 Introduction

Location-based services for mobile devices have achieved great popularity in recent times. Authentication, authorization, access control, accounting, and similar critical actions can be associated with the geographical locations of the devices. The location information is then used by service providers to provide diverse location-based services to the users [1]. However, unsecured location reporting mechanisms may have effects on trivial cases, such as, in social-games like FourSquare [2], and may even be of national security, as that of spoofing Drones with false location data [3].

A location proof for a user is verified with respect to the identity of the user, the location in question, and the time of visit. However, self-reported information regarding location presence can be easily spoofed. Global Positioning System (GPS) coordinates, cell triangulation in mobile phones, and IP address tracking are all susceptible to manipulation for making false location claims [4]. Conversely, automated location reporting violates users' privacy and introduces centralization bottleneck in the architecture [5].

There have been numerous proposals for user initiated location proof generation [1, 6–9]. The localization authority covering the area utilizes some secure distance-bounding mechanism to ensure the user's presence [10–12]. However, existing mechanisms overlook collusion attacks in their models. In a collusion attack, participating entities go into a mutual agreement and agrees to create counterfeit location proofs. Hence, the fake proofs resemble an actual proof and can be utilized by the user as a false evidence of presence at that location. The related works thus far have not considered any third-party endorsement for location proofs, which makes the schemes vulnerable to collusion attacks [1, 4–15].

This paper presents a distributed witness-oriented architecture for generating secure location proofs which is resistant to collusion attacks. The following scenario illustrates the practicality of a secure location proof mechanism.

A pharmaceutical agent travels around different places hoping to get a sale for the pharmaceutical company. Upon returning to office, he makes an expense claim for the money paid at the hotel where he was staying during the traveling period. However, in addition to the bill of receipt from the hotel, the company requires a proof of presence that the sales agent was actually residing at that specific hotel. Thus, the agent provides his secured proof of presence, which was collected from the hotel. The proof was also securely asserted by the hotel manager, or any other witness available at the hotel. The finance department of the company then validates the endorsed proof and pays the incurred cost to the sales agent.

Contributions. The contributions of this paper are as follows:

1. We have introduced a novel solution for obtaining distributed secure location proofs for mobile devices. The architecture allows generating witness-oriented asserted location proofs in a distributed environment which incorporates an additional endorsement by a third entity to ensure a collusion-resistant location proof.
2. We have presented an exhaustive security analysis of the proposed architecture against a detailed combinatorial study for collusion models and different attacks in the working protocol.
3. We have illustrated the feasibility of the proposed architecture for practical use using a proof-of-concept implementation. The prototype is used in an extensive experimental process to identify attacks and adjust the threshold values for the protocol.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 introduces the key terminologies and concepts in location proof systems. Section 4, discusses the potential attacks and challenges in a location proof generation scheme. We present our secure assertion oriented scheme in Section 5, and provide a security analysis in Section 6. The prototype implementation and simulation results have been presented in Section 7. Section 8 describes our experimental process to set the threshold values, and finally conclude in Section 9

2 Related Work

Location reporting mechanisms require a reliable and tamper proof architecture to preserve the integrity of the data. Traditional GPS systems are effective in general purpose location reporting [16]. However, it is not a suitable option in terms of security and indoor tracking. Recent papers have proposed a combination of GPS signals with

cellular tower triangulation and identifying the access network channel. Gabber *et al.* [17] utilized multi-channel information to verify the location. Unfortunately, malicious entities can bypass such combinatorial schemes [1, 8]. Additionally, GPS signatures [18] are not useful since they are open to spoofing attacks [8].

Hardware oriented localization techniques [19–21] measure signal attenuation and asynchronous measurement of round trip times to verify the presence of a certain user device in the vicinity [10, 22–25]. However, location reporting using signal attenuation can easily be manipulated by an attacker in close proximity of the devices. Furthermore, all of these mechanisms suffer from channel noise, limitations with line-of-sight, and complexity of deployment. In our design, we have considered a three-party interactive solution. We have used timing thresholds between each pair of communicating parties to ensure three-way proximity.

Collection of secure location proofs from a manager was discussed by Waters *et al.* [9]. Another approach for creating secure location proofs has been described by Saroiu *et al.* [1]. However, these schemes require highly coupled entities with a monolithically centralized architecture. Trusted platform module and virtual machine based attestation for trusted sensor readings have been proposed by Saroiu *et al.* [26] and Gilbert *et al.* [7] respectively. Luo *et al.* presented a method for obtaining privacy-preserved location proofs using a random nonce between the user and the provider [8]. Khan *et al.* presented a model for chaining location proofs in a chronological order for secure provenance [27].

Limitations of Current Research: In a free-to-act environment, participating entities may go into a mutual agreement and collude to produce a fake proof of presence. Furthermore, the person operating a mobile device may override certain operations on the device and manipulate the proofs. Such collusions between the parties can provide each other illegitimate benefits. Additionally, we consider the location authority to be possibly malicious as well. Given the location authority manipulates the proofs, most protocols that we have discussed so far will collapse. Furthermore, the flexibility and distributed mode of operation supported by such an architecture should be able to sustain the entropy, randomness, and falsified data generated by misbehaving entities in the environment.

3 Modelling a Secure Location Assertion

In this section, we present the notions of witnesses and assertions, the terminologies, and the models for creating and verifying secure location assertions.

3.1 Witnesses and Assertions

In everyday life, two parties considering each other as untrustworthy necessitates the involvement of a witness. In addition to the two parties involved in the information exchange, a witness provides a notarization of the statement. The notarized statement is then redistributed among the two parties, which bears the endorsement by the witness as an additional enforcement of the truth value of the content.

We utilize the same concept to create location proofs and have the proof asserted by a co-located witness. In this context, a witness is a spatio-temporally co-located entity with the user and the location authority. A witness will assert proofs only when willing to do so, and will not assert otherwise. Devices willing to assert location proofs sends

a registration request to the available location authority. In a commercially deployed scenario, the incentive of the witness can be based on awarded 'points' depending on valid assertions. The 'points' would add to the trust value of a witness and may be redeemed for membership benefits from the service provider. The assertions may also be used by the witness to prove co-location with the user. The witness can withdraw from the witness-list at any time by sending a withdrawal request to the location authority.

3.2 Terminologies

We have introduced certain terminologies in the description of our models, and also in designing the scheme for secure location proof assertion. A **User** U is a mobile entity that visits a location. The user is identified with a mobile device, which is used to determine his location and store the location proofs. A **Site** S is a physical region within a finite area under the coverage of one location authority. A **Location Authority** L is a stationery entity, which is responsible for providing location proofs for a particular site, and owns a unique identifier. A **Witness** W is a mobile user who can assert a location proof for the presence of another mobile device at a particular location. The presence of the witness does not imply an eye-witness, but rather a spacio-temporal co-location with the user at the site S. A **Witness List** WL provides the listing of all registered witnesses under the coverage of the location authority at a given time. Witnesses are registered against their cypto-ID. The witness list is preserved at the location authority and is used to provide the proof of witness's presence at the site. A **Crypto-Id** CID is a cryptographic identity for the user and witnesses, used in all phases of the protocol, ensuring privacy of the entities participating in the process. The users and witnesses will have the cypto-ID tagged with their certificates. A **Location Proof** LP is a token of evidence received by a user when visiting a specific site, and an **Asserted Proof** AP is a location proof LP asserted by a valid witness using his crypto-ID. Finally, an **Auditor** is an authority who is presented with an asserted location proof and confirms the legitimacy of the user's claim of presence at the particular site.

3.3 Threat Model

The two main targets considered in our threat model are the place and time of location proofs corresponding to a user. An adversary should not be able to create a proof for a location that the user has not visited, or a proof for a different time than the actual time of visit. An additional target is the identity and location privacy of users and witnesses. An attacker may create a dossier of users visiting a given location, and learn the location history and identities of other users it has encountered in the past. Unlike previous works [1, 8, 9], we assume the users, location authorities, or witness devices may be malicious and can collude with one another. It is assumed that the user has full access to the storage and computation of the device, can run an application on the device, and can delete, modify, tamper, or insert any content in the data stored on the device. The location authority or the user can create a puppet witness to produce false asserted proofs. We assume that no entities share their private keys at any point, and a three-way collusion scenario does not exist. We assume mobile devices are non-shareable private properties and the physical security of the phone depends on the user himself. Additionally, typical attacks such as MAC address fingerprinting are prevented via known techniques such

as MAC address cloning [28]. According to the protocol, we assume the presence of at least one witness at the given site who is willing to provide an assertion.

3.4 System Model

We assume the mobile devices carried by users are WiFi enabled. The devices have local storage for storing the proofs. A device visiting a site can find the location authority over the wireless network. It is assumed that the user, location authority, and witness can access each others' public key for a given Crypto-Id. The location authority periodically updates the available witness list. Witnesses are chosen at random for asserting a location proof. Upon completion of a schematic communication between the entities, the user is presented with a location proof and is stored on the user's device. At a later time, the user presents the location proof to an auditor.

4 Security and Challenges in Location Proofs

This section includes the fundamental security challenges which exist in any location proof protocol. We present the previous studies and illustrate the possible attacks in such distributed architectures for generating location proofs.

4.1 Challenges and Attacks

In our opinion, possibility of tampering with data in distributed flexible environments has a higher probability compared to any centralized architectures. Hence, we aim at making location assertions tamper-evident, assuming that all information are susceptible to tampering, as opposed to being tamper-proof. Therefore, we focus on ensuring detection of different types of attacks while generating a proof of presence. We list the potential attacks as follows.

False presence: A malicious user can create a fake location proof on his own, without being physically present at the location. The fake proof is supposed to resemble an actual proof, which the user could have actually collected from a valid location authority.

False timestamping *(backdating, future dating)*: In a backdating attack, the user and the location authority colludes to create a proof for a past time. Conversely, in future dating, the location authority and a user colludes to generate a proof with a future timestamp.

Implication: A location authority and/or a witnesses can falsely accuse a user of his presence at a certain location. In this case, the malicious location authority and witness colludes to generate a false proof of presence for the user.

False assertion: A user can collude with a witness, and generate a falsely asserted location proof. The truth value in such a fake proof is reinstated with the assertion received from the other user.

Denial of presence: A user can visit a location and at a later time, deny his presence at that location. In such a case, the user actually denies the validity of a certain location proof that has been been generated upon his presence at that particular location.

Proof switching: The user is expected to have full access to all storage facilities on his mobile device. Hence, the user utilizes the legitimate proof and manipulates the information to create a false proof for a different location.

Relay attack: A user can use a proxy to relay the requests and collect a location proof. Alternatively, a location authority can maliciously relay assertion requests with the witness not being present at the site.

Sybil attack: A Sybil attack occurs when a single user generates multiple presence and identities [29]. A user can launch a Sybil attack by generating multiple identities representing a user and a witness and provide false endorsements for location proofs.

Denial of witness's presence: At the time of proof verification, the user can claim the absence of witnesses at the site or falsely claim an assertion to be counterfeit. The user and the location authority may also collude and claim the non-availability of witnesses.

Privacy violation: An attacker may capture an asserted location proof generated for a user, and discover the identity of the user and/or the witness.

5 A Secure Location Proof Assertion Scheme

In this section, we present the design, schematic definitions, architecture and the protocol for a secure location proof assertion scheme.

5.1 Schematic Description of Secure Location Proof Assertion

In this section, we define the schematic description of each message in all steps of the protocol, in sequence of their occurrences. Initially, the user *U*, sends a proof request, *PReq*, to the location authority, *L*.

$$pReq =< CID_{\mathrm{U}}, t_{\mathrm{U}} > \tag{1}$$

Here, in expression 1, CID_U is the cryptographic identity of the user *U*, and t_U, is the timestamp from the user *U*'s mobile device. To state that, user *U* has visited a site with location authority identifier *L*, at time t_L – the current time at the location authority *L*, the location authority prepares a location statement *LS* as follows:

$$LS =< CID_{\mathrm{U}}, L, t_{\mathrm{L}} > \tag{2}$$

Hence, the location authority creates a location proof *LP*, to be sent to the user *U*, using the location statement *LS*, formed in expression 2. Additionally, the location authority *L* also forms the assertion request *AReq* for *LP* to be sent to witness *W*.

$$LP = AReq =< LS, S_{\mathrm{L}}(LS) > \tag{3}$$

Here, in expression 3, $S_L(LS)$ represents the digital signature computed on location statement *LS*, from expression 2, using the location authority's private key. Thus, the location proof *LP* is sent to user *U*, and an assertion request *AReq* is sent to the witness *W*. Next. the asserted statement *AS* is created by a witness *W* to assert the *AReq*. The assertion statement is prepared as follows:

$$AS =< CID_{\mathrm{W}}, CID_{\mathrm{U}}, L, h(LP), t_{\mathrm{W}} > \tag{4}$$

In expression 4, CID_W and CID_U are the cryptographic identifiers for the witness *W*, and the user *U* respectively. Additionally, t_W is the signed asserted timestamp from the witness' mobile device. The witness includes *h(LP)*, a cryptographic hash of the *LP*. Subsequently, the witness *W* prepares an assertion *A*, as shown below.

$$A =< AS, S_{\mathrm{W}}(AS) > \tag{5}$$

The assertion in expression 5 includes an asserted statement *AS* from expression 4, and $S_W(AS)$ is a signature computed by the asserting witness *W* on *AS*. Thus, an asserted

proof of presence at site *S*, created by the witness *W*, is a pair of values: the location proof *LP*, and the assertion *A*. The asserted location proof, *ALP* is defined as thus:

$$ALP =< LP, A > \tag{6}$$

The user *U* receives *ALP* as shown in expression 6, and issues a verification request *VReq* to be sent to the witness *W* as follows:

$$VReq =< ALP, LP, h(ALP, LP), t_u > \tag{7}$$

In expression 7, the user *U* had already received the location proof *LP* (expression 3) and the asserted location proof *ALP* (expression 6). The user includes a signed timestamp t_u for the current time on the user's device, and *h(ALP,LP)*, a cryptographic hash function on both the location proof *LP* and the asserted proof *ALP*. The verification response *V*, sent by the witness *W* is defined as:

$$V =< R, t_{\mathrm{WV}} > \tag{8}$$

Here, $R \in \{YES, NO\}$, and $\mathrm{t_{WV}}$ is the response timestamp for the witness verification from the witness' mobile device. The verification statement *VS* is thus defined as follows:

$$VS =< V, S_{\mathrm{W}}(V) > \tag{9}$$

In expression 9, *V* is the verification response from expression 8, and $S_W(V)$ is a signature computed by the witness *W* on *V*. Finally, the acknowledgement *ALPAck* is created by user *U* and sent to the location authority *L* as follows:

$$ALPAck =< S_{\mathrm{U}}(LP, AS), h(LP, AS), t_{\mathrm{t}} > \tag{10}$$

The acknowledgement *ALPAck* shown in expression 10 includes $S_U(LP,AS)$, a cryptographic signature from user *U*, on the location proof *LP* and the assertion statement *AS*. This is then sent to the location authority *L*, to be stored, as a receipt for the asserted location proof received by the user *U*.

5.2 Location Assertion Protocol Architecture

In our proposed architecture, we assume that each entity is registered with a service provider. Users, witnesses, and location authorities register with the centralized system, with a unique identification criteria, such as the Social Security Number, passport number, driving license, and trade license. The entities will get a crypto-ID tagged with a certificate containing the public/private key-pair. This is the only component which requires a centralized mode of operation. However, we claim that this is a one-time procedure, and does not constitute any obstruction as a bottleneck in rest of the protocol. Secondly, there exists a mechanism to distribute the public certificates for all the entities. The user *U*, witness *W*, and the location authority *L*, should be able to collect each other's public-key certificates. Finally, it is given that all communications between the user *U*, the location authority *L*, and the witness *W*, take place over secure socket layer (SSL) connections and public key encryption. The sequence of interaction for creating an asserted location proof is illustrated in figure 1 and is described as below.

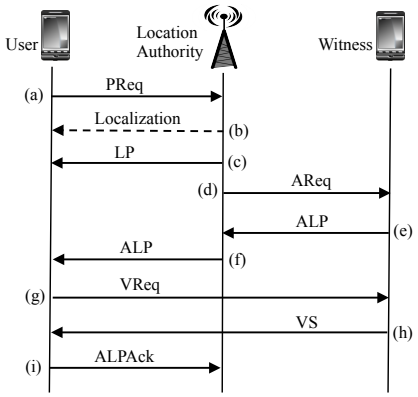


Fig. 1: Sequence Diagram for the Location Assertion Protocol

(a) **Location authority discovery and proof request:** Each location is identified by a unique global identifier and are publicly available (via a lookup), or that the location authorities periodically broadcast their information on the local network. The user obtains the identity of the location authority and sends a location proof request *PReq* to the location authority *L*, as shown in expression 1.

(b) **Secure localization:** Upon receiving the *PReq* message, the location authority runs a secure localization step to determine whether the device is actually present there.

(c) **Location proof generation:** The location authority *L* generates the location proof *LP*, as shown in expression 3, and sends it to the requesting user.

(d) **Proof assertion request:** The location authority *L* has a witness list *WL* consisting of the available witnesses willing to serve for asserting location proofs. The location authority *L* sends an assertion request *AReq*, as shown in expression 3, to a randomly selected witness *W* from the witness list *WL*.

(e) **Asserted message creation:** The witness *W* receives the assertion request *AReq* and verifies the location statement *LS* included within *AReq*. Upon a successful verification of all information, the asserted location proof *ALP*, as shown in expression 6, is sent to the location authority *L*.

(f) **Assertion verification and relay:** The location authority *L* receives and verifies the asserted location proof *ALP*. The location authority *L* verifies the time lapse between sending an assertion request *AReq* and receiving the asserted location proof *ALP*, i.e., difference between t_L available from *ALP* and the current time at the location authority *L*. A maximum threshold for the time difference is enforced to detect any proxy forwarding delay by the witness. Upon successful verification, the location authority *L* relays the asserted location proof *ALP* to the user *U*.

(g) **Verification request:** Once the user *U* has received both the location proof *LP* and the asserted location proof *ALP*, he sends the verification request *VReq* directly to the witness, as shown in expression 7.

(h) **Verification response:** The witness *W* receives the verification request *VReq* and validates the assertion provided earlier. The witness calculates the difference between the time t_W, available in the assertion statement *AS* (expression 4), with the current time on the witness device. An acceptable threshold for the time difference ensures that the user is not a proxy relay attack. If successful, the witness *W* creates a verification statement *VS*, as shown in expression 9, and sends it to the user *U*.

(i) **Location proof receipt:** Finally, user *U* receives the verification statement *VS* from the witness *W*. The user *U* verifies the difference between the time in the verification request t_u, and the current time on the user's device when it receives the verification response. A maximum threshold for the delay ensures that the witness is not proxying the assertion and the verification requests. Once verified, the user creates an acknowledgement *ALPAck*, as shown in expression 10, and sends it to the location authority *L*. The user *U* then stores the asserted location proof *ALP* on his device for the specific site *S*, and hence, completes the protocol.

Subsequently, the location authority *L* stores the receipt for the location proof and maintains a publicly visible list of these tickets. At every epoch, it publishes the current state of this list along with a signature. The published list is used to prevent back-dating and future-dating attacks.

6 Security Analysis

In this section, we present an analysis of the security properties of our schemes. We start by enumerating the different types of attackers and combination of collusions among the existing entities. Furthermore, we analyse how our scheme can protect against attacks, which are possible in such colluded environments.

6.1 Collusion Patterns

We define the following symbols: honest user *U*, malicious user $\bar{U}$, honest location authority *L*, malicious location authority $\bar{L}$, honest witness *W*, and a malicious witness $\bar{W}$. The eight possible combinations for collusion patterns and the corresponding attacks are shown in table 1. The protocol ensures mutual communication among all entities. Thus, any collusions leading to a fake proof generation can be easily identified by the valid entity at specific stages of the protocol. A thorough analysis on each collusion pattern is presented in the following sub-section.

Notation	**Attack**
U L W	No collusion.
$\bar{U}$ L W	False proofs, reordering, denial of presence, proof switching, relay attack.
U $\bar{L}$ W	Denial of service, implication.
U L $\bar{W}$	False endorsement, privacy.
U $\bar{L}\bar{W}$	Implication, relay attack, replay attack.
$\bar{U}$ L $\bar{W}$	False endorsement, relay attack, Sybil attack.
$\bar{U}\,\bar{L}$ W	False proofs, relay attack, replay attack.
$\bar{U}\,\bar{L}\bar{W}$	False proofs.

Table 1: Collusion Models and Corresponding Threats

6.2 Threat Analysis

We have made a thorough security analysis of all the possible combinations of the user, location authority, and the witness. Table 1 summarizes the different attack scenarios and the corresponding threats.

[ULW] All honest entities do not imply a threat of generating false location proofs.

[$\bar{\mathbf{U}}$LW] A malicious user $\bar{U}$ can request false location proofs. However, if the location authority *L* and witness *W* are honest, this attack does not succeed. An honest location authority *L* will not sign a false location proof. Additionally, an honest witness *W* will not endorse a location statement which is not accompanied by a proof from the location authority *L*. In case of a relay attack, the proxy forwarding delay can be detected in step (h) of the protocol, and thus can be rejected.

[U$\bar{\mathbf{L}}$W] The dishonest location authority $\bar{L}$ will never have the final receipt from the user *U* and thus cannot create a false proof. The honest witness *W* will also not assert a location proof, unless it can detect user *U*'s presence. The malicious location authority $\bar{L}$ may provide a false timestamp. However, an honest witness will not endorse a proof if the timestamp differs a lot from its own timestamp. Additionally, any illegitimate information by the malicious location authority will force the user *U*, or the witness *W* to forfeit the asserted location proof protocol.

[UL$\bar{\mathbf{W}}$] A malicious witness $\bar{W}$ cannot do any harm, other than denial of service and privacy violation of the user *U*. However, the cryptographic identity of the user CID_U does not allow the malicious witness $\bar{W}$ to reveal the user's actual identity. Furthermore,

a falsely asserted location proof will be discarded by the location authority *L*, before the location authority *L* relays the asserted location proof to the user *U*.

[U$\bar{\mathbf{L}}\bar{\mathbf{W}}$] A malicious location authority $\bar{L}$ can collude with a dishonest witness $\bar{W}$ and create false location proofs for a user. However, if the user never participated in a proof protocol with the location authority, such an attack will not work. The malicious location authority $\bar{L}$ can give a user a backdated or a future dated timestamp. Subsequently, a colluding malicious witness $\bar{W}$ can endorse such a false timestamped proof. However, the user *U* finally verifies the location proof and the asserted location proof, and has the option of discarding the protocol by not sending the final receipt for the asserted location proof. A relay attack can also be identified by the user *U* between step *(h)* and step *(i)*. The malicious location authority $\bar{L}$ also has the option for storing a previous proof, endorsed by a valid witness *W*, and use it later to launch a replay attack. However, the user *U* directly communicates with the witness during endorsement verification. Thus, in case of any discrepancy with the timing threshold, the user *U* can discard the proof completely.

[$\bar{\mathbf{U}}$L$\bar{\mathbf{W}}$] A malicious user $\bar{U}$ and a colluding witness $\bar{W}$ cannot create falsely asserted location proofs. The location authority *L* denies to cooperate with the dishonest user $\bar{U}$ and the witness $\bar{W}$, based on the comparison of the timestamps t_U and t_W, or any invalid information included in the process of asserting the location proof. A Sybil attack is also possible in this case. However, the centralized registration system, a requirement of the architecture, prevents a user $\bar{U}$ to create a witness profile $\bar{W}$ on the same device. Additionally, the location authority warrants for witness devices which are already registered at the location authority *L*. A relay attack with a proxy user $\bar{U}$ and a proxy witness $\bar{W}$ is also detected by the location authority *L* in steps *(b)* and *(f)* respectively.

[$\bar{\mathbf{U}}\bar{\mathbf{L}}$W] A malicious user $\bar{U}$ and a dishonest location authority $\bar{L}$ can collude to create a false proof with backdated or future-dated timestamp. The falsely created asserted location proof can be utilized to launch a relay and a replay attack. However, an honest witness *W* will not endorse a false proof with an incorrect timestamp. In step *(h)*, from the time difference between t_W and the current time on the device, the witness can successfully identify a relay or a replay attack and will refrain from sending a positive response to the user $\bar{U}$ in the verification phase.

[$\bar{\mathbf{U}}\bar{\mathbf{L}}\bar{\mathbf{W}}$] A three way collusion is not considered in our scheme. However, a backdated attack can be detected by the auditor if checks the published accumulator by the location authority for the epoch corresponding to the proof timestamp. The only attack that is possible here is a post-dating attack, when the user $\bar{U}$, location authority $\bar{L}$, and witness $\bar{W}$, collude to create a location proof with a future timestamp, and $\bar{L}$ does not publish it in the epoch report. However, we claim that any distributed security protocol without centralized monitoring requires at least one entity which is valid. Hence, the successful completion of any security protocol is protected against the legitimate entity, who plays the role of the situational verifier. Nonetheless, an auditor may impose a stricter proof model involving asserted location proof statements from multiple closely located location authorities to verify the actual presence.

7 System Evaluation

In this section we present the performance of the designed architecture illustrated utilizing a prototype implementation. Additionally, we present a comparative analysis of our protocol against a similar protocol proposed by Luo *et al*. [8].

7.1 Protocol Implementation

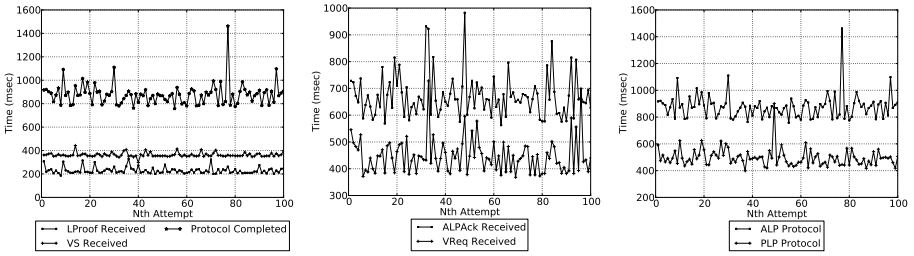
To evaluate the feasibility and performance of the protocol, we developed the prototype applications for location authority, user and witness. Applications for user and witness were built on the Android platform. In our experiment, for simluating a user, we used a *HTC Evo 4G* smart phone with *Android 2.3.3* operating system. The witness was simulated on a *Motorola XT875* smart phone, with *Android 4.0.3* operating system. The location authority was implemented as a desktop application, using *JDK 1.6*, and ran it on *OSX 10.8.2*, equipped with *Intel Core i5 1.7 GHz* processor and *4GB 1600 MHz DDR3* RAM. For the testbed network, we used WiFi for communication among the different entities in the protocol, and generated the asserted location proofs. We used the RSA (2048 bit) for generating signatures and encryption of the packets. Additionally, we used SHA-256 for generating the hash values. We assume that all the three entities have access to each other's public keys. Hence, the processing delay does not include the time to access the key over the network.

7.2 Performance Analysis

We evaluated the performance of three important steps of the protocol from the user application. We recorded the timestamps at different phases of the protocol for 100 complete execution cycles. Initially, we recorded the time lapsed after sending a proof request *PReq* to the location authority *L*, and eventually receiving a location proof *LP*. We denote this as *LProof Received* time. Subsequently, we recorded the time lapsed between sending the verification request *VReq* and receiving the verification statement *VS* from the witness *W*, which is denoted here as *VS Received* time. Finally, we measured the time required to complete the whole protocol. Figure 2a represents the time required for each step in every iteration, and figure 3 illustrates the average time required for the individual steps.

In our proposed scheme, the mean time for *LProof Received* and *VS Received* were 228 milliseconds and 362 milliseconds respectively. Although the computation needed for generating the location proof *LP* and the verification statement *VS* is similar (generating the packet then signing it), the *VS Received* time is higher than the *LProof Received* time. This behavior is natural, as the witness's device has less computation power than the location authority's device.

In the protocol, the location authority *L* forwards the asserted location proof *ALP* to the user *U*. In the end, the location authority *L* receives the acknowledgement *ALPAck* receipt from the user *U*. We measured the time required between these two steps. The time measurement is noted as *ALPAck Received* on figure 2b, which depicts this processing delay for each iteration. Additionally, the average time required for *ALPAck Received* is shown in figure 3.



(a) User Application (b) Witness & Location Authority (c) Comparison between ALP & PLP

Fig. 2: Protocol Performance Evaluation

The witness *W*, sends the asserted location proof *ALP* to the location authority *L*. The witness then *W* waits for the verification request *VReq* from the user. The time required between sending the asserted location proof *ALP* and receiving the verification request *VReq* is important from the witness's point of view. We measured the processing delays between these two steps, which is denoted as *VReq Received*. Figure 2b illustrates the time for each iteration on the witness's device, with the average time shown in figure 3.

7.3 Performance Comparison

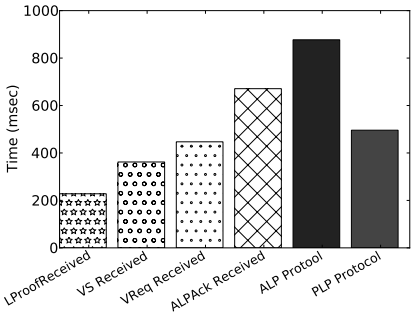


Fig. 3: Average Time Required for Different Steps of the Protocol

To perform a comparative analysis of our proposed protocol (Asserted Location Proof or *ALP* protocol), we selected another secure location proof protocol, namely 'Proactive Location Proof' or *PLP* protocol, proposed by Luo *et al*. in [8]. Both the protocols were compared based on their time of completion for receiving the location proof. Figure 2c illustrates the time required to complete each protocol. The average processing time for the *ALP* protocol is 877 milliseconds, whereas that of the *PLP* protocol is 496 milliseconds. Given the fact that we have more phases in our protocol including numerous encryption and decryption operations, the *ALP* should be taking a longer processing time. However, the comparison demonstrates that the processing time for ALP is still comparable to rather simplistic models like PLP. Additionally, none of the other protocols so far have neither considered collusion attacks nor the presence of malicious location authority.

The results above show some overhead processing in our proposed protocol. However, it provides an extra level of security by getting the assertion from the witness, hence adding to the trust value of such location proofs. The completion time for the protocol is still less than 1 second, which is a reasonable latency for practical usage. Addition of the witness increases the attack surface for the protocol. Nonetheless, we have proved in section 6 that our proposed protocol is resilient to all combinations of collusion attacks.

8 Threshold Adjustment

We applied a practical approach to determine the thresholds in different phases of the asserted location proof protocol. Resistance against relay attacks have been illustrated using the following experimental setup, where we adjust the optimal values for the threshold according to the requirements of the system.

8.1 Threshold Initialization

We need three threshold values to identify relay attacks in the protocol. At first, in step *(f)* of our protocol, the location authority *L* verifies the time lapse between sending *AReq* and receiving *ALP*, $\Delta(LA_{AReq-ALP})$, using the threshold T_{LW} to identify a proxy witness. In step *(h)*, the witness measures the required time between sending *ALP* and receiving *VReq*, $\Delta(W_{ALP-VReq})$, and compares this time with the threshold T_{WU} to detect the presence of a proxy user. Finally, in step *(i)*, the user measures the elapsed time between sending *VReq* and receiving *VS*, $\Delta(U_{VReq-VS})$, and compares it with the threshold T_{UW} to identify the presence of a proxy witness. The initial threshold value is set as the mean time of completion for the given phases of the scheme, which was calculated from 100 experimental executions of the protocol. Subsequently, the threshold was set at that value to detect the presence of proxy users or witnesses. The mean values for $\Delta(LA_{AReq-ALP})$, $\Delta(W_{ALP-VReq})$, and $\Delta(U_{VReq-VS})$ are thus set to values for T_{LW}, T_{WU}, T_{UW} at 240.754, 552.464, and 346.004 milliseconds respectively.

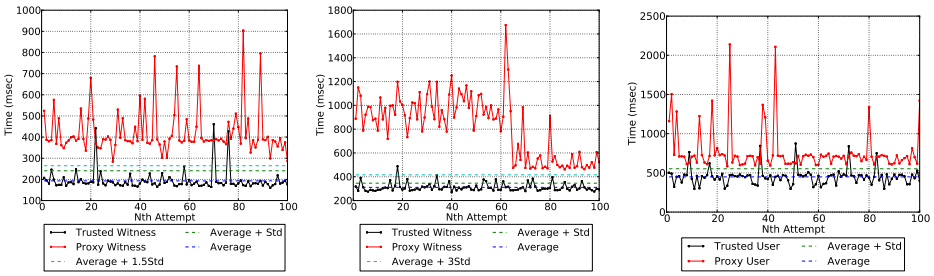
8.2 Variable-Distance Threshold Measurements

The next phase of the work included a variable-distance setup for the protocol. The recorded times were used to justify the values for T_{LW}, T_{UW}, and T_{WU}. We placed the user *U*, witness *W*, and the location authority *LA* at varying distances and recorded the time measurements for each of $\Delta(U_{VReq-VS})$, $\Delta(LA_{AReq-ALP})$, and $\Delta(W_{ALP-VReq})$. The recorded times for each of the values for varying distances are shown in table 2.

Case	Dist. (ft)	Time (ms)
U - W	33	381.093
$\Delta(U_{VReq-VS})$	62	383.232
	80	383.232
	100	453.131
W - LA	3	235.178
$\Delta(LA_{AReq-ALP})$	30	251.032
	50	230.801
	70	349.787
U - LA	36	487.437
$\Delta(W_{ALP-VReq})$	93	512.459
	110	2120.656
	132	1949.892

Table 2: Variable-Distance Measurements

The recorded times show that the time intervals tend to increase as the distance between the entities are increased. Additionally, we observed that the previously set values for the thresholds do not suffice the purpose of determining the proxy attacks in each of the cases. Therefore, the next phase for adjusting the threshold included performing a relay attack using a proxy witness and user. Subsequently, we utilized the measurements from the relay attack to adjust our threshold values using a sliding threshold model.



(a) T_{LW} (b) T_{UW} (c) T_{WU}

Fig. 4: Threshold Identification

8.3 Relay Attacks using Proxy

The value of the thresholds have been determined using relay attacks. We executed a relay attack using a proxy to forward messages between two networks. In the first case, we utilized a proxy to relay packets to a remote network to a witness which is not not spatially co-located with user and location authority. We calculated the time lapse between sending *VReq* and receiving *VS* for this attack scenario. The recorded T_{LW}'s and T_{UW}'s for both trusted and proxy witnesses are shown in figure 4a and figure 4b respectively. Next, we performed a similar experiment using a user proxy. The proxy was present to relay the packets to the user on the remote network. The recorded T_{WU}'s for both the trusted and proxy users are shown in figure 4c.

8.4 Sliding Threshold Model

We utilized a sliding threshold model to determine optimal values for T_{LW}, T_{UW}, and T_{WU}. Initially, we started with a minimum value to specify the optimal threshold, and observed the percentage of attacks successfully identified. Additionally, we also calculated the percentage of false alarms when the threshold is set at the given value.

For determining an optimal T_{UW}, we set the initial threshold at [Mean (μ) + Standard Deviation (σ)]. The threshold for T_{UW} was thus at 552.46 milliseconds, and the corresponding attack identification and false alarm was found to be at 100% and 6% respectively.

The threshold was set at different incremental values to reduce the false alarm rate in the protocol. The next experimental threshold was set at $[\mu + 1.3\sigma]$, where the attack identification was still at 100%, while the false alarm had dropped to 14%. With a gradual increase of the threshold, we saw no decrease in the percentage of attacks identified, but the false alarm rate reduced to 5% by the time we reached $[\mu + 2\sigma]$. With T_{UW} set at $[\mu + 3\sigma]$, the attack identification was still 100% but the false alarms has reduced to only 1%. As we increased the threshold beyond $[\mu + 3\sigma]$, we observed the false alarm rate reduced to the point where it was still 1%, and the percentage of attack identification had started to drop. At this point, the sliding threshold was thus fixed at $[\mu + 3\sigma]$ for T_{UW}. The values from our simulation has been summarized in table 3.

We applied the same sliding threshold model to determine the threshold values for T_{LW} and T_{WU} respectively. Upon similar experimental evaluations as above, the optimal threshold for T_{LW} has thus been set at $[\mu + 1.5\sigma]$, with a value of 264.7 milliseconds. The corresponding attack identification and false alarm rates are 100% and 3% respectively.

Similarly, the optimal threshold for T_{WU} has been found to be at $[\mu + \sigma]$ with a value of 552.46 milliseconds. The corresponding attack identification and false alarm rates are 100% and 6% respectively. The results from the sliding threshold model for T_{LW} and T_{WU} in presented in table 3.

Threshold	Step	Value	(%)Attack Detection	(%)False Alarm
T_{WU}	$\mu + \sigma$	**552.46**	100	6
	$\mu + 1.3\sigma$	583.50	99	6
	$\mu + 1.5\sigma$	604.20	93	6
	$\mu + 1.7\sigma$	624.89	73	5
T_{LW}	$\mu + \sigma$	240.8	100	6
	$\mu + 1.3\sigma$	255.14	100	4
	$\mu + 1.5\sigma$	**264.7**	100	3
	$\mu + 1.8\sigma$	279.04	100	3
	$\mu + 2\sigma$	286.5	98	3
T_{UW}	$\mu + \sigma$	346	100	15
	$\mu + 1.5\sigma$	363.5	100	10
	$\mu + 1.8\sigma$	374	100	8
	$\mu + 2\sigma$	381	100	5
	$\mu + 2.3\sigma$	391.5	100	3
	$\mu + 2.5\sigma$	398.5	100	2
	$\mu + 3\sigma$	**416.01**	100	1
	$\mu + 3.5\sigma$	433.51	100	1
	$\mu + 4.3\sigma$	461.51	98	1

Table 3: Justification of Threshold Values

9 Conclusion

Collection and verification of location proofs have significant real-life applications in location-based services. In this paper, we introduced a novel architecture for obtaining secure asserted location proofs from a location authority and a spatio-temporally co-located witness. Our design is set up on a tamper-evident platform, in contrast to a tamper-proof concept. We illustrate how our proposed scheme provides defense against all possible forms of attacks and collusion models within the entities. Currently, we are developing an algorithm for multi-metric selection of witnesses from the list of currently registered witnesses based on their trust values in the protocol. For further research, we will extend the protocol and implement a granular concept of information visibility to preserve user privacy. We are also working on secure location provenance chains to allow auditors to validate the user's order of presence at different locations.

Acknowledgement

This research was supported by a Google Faculty Research Award and the Department of Homeland Security Grant #FA8750-12-2-0254.

References

1. S. Saroiu and A. Wolman, "Enabling new mobile applications with location proofs," in *Proc. of HotMobile*, 2009, pp. 1–6.
2. J. VanGrove, "Foursquare cracks down on cheaters." Online at http://mashable.com/2010/04/07/foursquare-cheaters/, April 2010.
3. I. Maduako, "Wanna hack a drone? possible with geo-location spoofing!" Online at http://geoawesomeness.com/?p=893, 26th July 2012.
4. N. O. Tippenhauer, K. B. Rasmussen, C. Popper, and S. Capkun, "iPhone and iPod location spoofing: Attacks on public WLAN-based positioning systems," *SysSec Technical Report, ETH Zurich*, April, 2008.

5. A. J. Blumberg and P. Eckersley, "On locational privacy, and how to avoid losing it forever," Online at https://www.eff.org/wp/locational-privacy, August 2009.
6. B. Davis, H. Chen, and M. Franklin, "Privacy-preserving alibi systems," in *Proc. of ASIACCS*. ACM, 2012, pp. 34–35. [Online]. Available: http://doi.acm.org/10.1145/2414456.2414475
7. P. Gilbert, L. P. Cox, J. Jung, and D. Wetherall, "Toward trustworthy mobile sensing," in *Proc. of HotMobile*. ACM, 2010, pp. 31–36.
8. W. Luo and U. Hengartner, "Proving your location without giving up your privacy," in *Proc. of HotMobile*, 2010, pp. 7–12.
9. B. R. Waters and E. W. Felten, "Secure, private proofs of location," Technical report TR-667-03, Princeton University, January 2003.
10. S. Brands and D. Chaum, "Distance-bounding protocols," in *Proc. of EUROCRYPT*. Springer-Verlag New York, Inc., 1994, pp. 344–359.
11. J. T. Chiang, J. J. Haas, and Y.-C. Hu, "Secure and precise location verification using distance bounding and simultaneous multilateration," in *Proc. of WiSec*. ACM, 2009, pp. 181–192.
12. K. B. Rasmussen and S. Čapkun, "Realization of RF distance bounding," in *Proceedings of the USENIX Security Symposium*, 2010.
13. A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing," in *Proc. of NDSS*, 2011.
14. P. Traynor, J. Schiffman, T. La Porta, P. McDaniel, and A. Ghosh, "Constructing secure localization systems with adjustable granularity using commodity hardware," in *Proc. of GLOBECOM*, 2010, pp. 1–6.
15. J. Brassil, R. Netravali, S. Haber, P. Manadhata, and P. Rao, "Authenticating a mobile device's location using voice signatures," in *Proc. of WiMob*. IEEE, October 2012, pp. 458 –465.
16. P. Enge and P. Misra, "Special issue on global positioning system," *Proceedings of the IEEE*, vol. 87, no. 1, pp. 3 –15, jan. 1999.
17. E. Gabber and A. Wool, "How to prove where you are: tracking the location of customer equipment," in *Proc. of ACM CCS*. ACM, 1998, pp. 142–149.
18. D. E. Denning and P. F. MacDoran, "Location-based authentication: Grounding cyberspace for better security," *Computer Fraud & Security*, vol. 1996, no. 2, pp. 12–16, 1996.
19. S. Capkun and J. Hubaux, "Secure positioning of wireless devices with application to sensor networks," in *Proc. of INFOCOM*, vol. 3. IEEE, 2005, pp. 1917–1928.
20. N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in *Proceedings of the 2nd ACM workshop on Wireless security (WiSe)*. ACM, 2003, pp. 1–10.
21. S. Čapkun and M. Čagalj, "Integrity regions: authentication through presence in wireless networks," in *Proc. of ACM WiSe*. ACM, 2006, pp. 1–10.
22. Aruba Networks, Inc., "Dedicated air monitors? you decide." Online at http://www.arubanetworks.com/technology/tech-briefs/dedicated-air-monitors/, 2006.
23. S. Pandey, F. Anjum, B. Kim, and P. Agrawal, "A low-cost robust localization scheme for wlan," in *Proc. of WICON*. ACM, 2006, p. 17.
24. P. Tao, A. Rudys, A. Ladd, and D. Wallach, "Wireless lan location-sensing for security applications," *Computing Reviews*, vol. 45, no. 8, pp. 489–490, 2004.
25. M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala, "Pinpoint: An asynchronous time-based location determination system," in *Proceedings of the 4th international conference on Mobile systems, applications and services*. ACM, 2006, pp. 165–176.
26. S. Saroiu and A. Wolman, "I am a sensor, and i approve this message," in *Proc. of HotMobile*, 2010, pp. 37–42.
27. R. Khan, S. Zawoad, M. Haque, and R. Hasan, "OTIT: Towards secure provenance modeling for location proofs," in *Proc. of ASIACCS*. ACM, 2014.
28. I. Martinovic, F. Zdarsky, A. Bachorek, C. Jung, and J. Schmitt, "Phishing in the wireless: Implementation and analysis," in *Proceedings of IFIP SEC*, 2007, pp. 145–156.
29. J. Douceur, "The Sybil attack," *Peer-to-peer Systems*, pp. 251–260, 2002.