I am Alice, I was in Wonderland: Secure Location Proof Generation and Verification Protocol

Chitra Javali*†, Girish Revadigar*†, Kasper B. Rasmussen‡, Wen Hu*, and Sanjay Jha*
*School of Computer Science and Engineering. UNSW Australia, Sydney, Australia
†DATA61|CSIRO, Sydney, Australia
‡University of Oxford, Oxford, UK
Email: {chitra.javali, girish.revadigar, wen.hu, sanjay.jha}@unsw.edu.au, kasper.rasmussen@cs.ox.ac.uk

***Abstract*—In recent years, the proliferation of wireless devices has contributed to the emergence of new set of applications termed as Location Based Services (LBS). LBS provide privileges to mobile users based on their proximity to a facility. In order to gain benefits, users may lie or falsely claim their location. Hence, it is essential to verify the legitimacy of users. In this paper, we propose our novel solution for generating location proof for mobile users and verification of the location claim by application services. Our protocol exploits unique Wi-Fi signal characteristics and employs an information theoretically secure fuzzy vault scheme. We provide a detailed theoretical and experimental evaluation of our protocol. Our solution is faster by an order of magnitude, and the performance of our scheme is independent of the location tag size and distance between the mobile user and location proof provider compared to the state-of-the-art.**

***Index Terms*—Location proof, Location based services (LBS), Wireless channel characteristics, Fuzzy vault.**

I. Introduction

The proliferation of mobile devices has led to the rapid increase in various location based applications and services. Location based applications such as Foursquare [2], Yelp [3], etc., award incentives to mobile users who check-in most frequently at a particular location. Additionally, the owner of the location also rewards mobile user with gift vouchers. These location based applications/services (LBS) can be further applied to other access control systems where proximity detection is required. For instance, in a critical health-care system, the authorized personnel e.g., doctors may be allowed to access the complete information/records of a patient if present in/around the vicinity of a hospital, else may have limited access to the records. Similarly, for entertainment applications, an online movie downloading system may provide content only to the users if present in/around an area of interest, and surcharge other customers present at a different location/place. In such scenarios, it is important for the user to prove his/her location to the services.

In all the above applications, in order to gain benefits the user may lie about his/her location. For e.g., a customer may falsely claim that he/she was in the shop at a particular time, a hospital authority may lie about his/her location to gain access to a celebrity patient's records. Current LBS like Foursquare use GPS to verify the legitimacy of the user. However, using GPS for verification has several drawbacks as it has limited signal coverage in indoor environments, and the second factor is that it can be manipulated by a user. A recent analysis by researchers on Foursquare check-in traces illustrates that there exist a large amount of forged or superfluous location information uploaded by mobile users [32]. The solution to address this problem is *Location proof*, the data containing the location information and time stamp provided by a trusted entity, say a wireless infrastructure e.g., Access Point (AP) to the mobile user, which can be later verified by LBS for legitimacy to grant privileges. For example, in a healthcare application, when a user tries to access a database record, the server may request for a location proof. The user may subsequently request a nearby AP to provide a proof which he/she would forward to the server. The server can then grant or deny access based on the legitimacy of the proof submitted.

In this paper, we propose a location proof generation and verification scheme which *proves* the presence of the user within an area of interest at a particular time and the claim is *securely verifiable* by LBS. Our mechanism is based on the unique wireless channel characteristics, i.e., channel state information (CSI) [9] and fuzzy vault, a cryptographic primitive [10], [11], [25]. CSI is the fine grained physical layer information obtained from 802.11 Wi-Fi packet traces [4]. Fuzzy vault is an information theoretically secure scheme in which a user can hide his/her information by a feature set. Another user can recover the information only if his/her feature set sufficiently overlaps with the set used to hide it.

The foremost challenges in a location proof system are to ensure that (i) it is infeasible for users to modify the issued location proofs, (ii) the services must be able to identify fake location proofs submitted by dishonest users, and (iii) it is deployable with existing infrastructure. Our scheme addresses the above design challenges and provides additional security by segregating the generated location tag into two feature sets. A user holds only a part of the location proof and it is infeasible to manipulate the proof unless the corresponding feature set is combined with it. We show that it is computationally infeasible for an adversary to know the location tag without being aware of another set. Moreover, our solution also helps to obtain additional information about the user's context by validating the approximate path traversed by the user in an indoor environment. The simplest method to obtain a location proof is to request a nearby AP to issue and forward it to LBS. However, in such scenarios the user cannot claim the proof at

a later point of time. If the user just had to claim a location to the service, anyone would be able to claim popular locations (or check the locations of other people). Our *Location proof* contains information about the channel conditions at the time the user was at the location, which makes it hard to forge.
Our contributions:

- We design a novel location proof generation and verification mechanism for mobile users to claim their proximity to a facility. Our solution is the first to leverage CSI, the unique spatio temporal characteristics of wireless channel and fuzzy vault technique to generate and verify the location proof.
- We provide a detailed theoretical and experimental analysis of our proposed protocol. We conduct an extensive set of experiments in multiple real indoor environments with commercially available off-the-shelf devices to evaluate the performance of our proposed protocol.
- Our results reveal that in the proposed system, (i) the location proof cannot be replicated by any adversary even if present in the vicinity of the legitimate user, (ii) the location proof cannot be transferred, modified, or reused by the same user to claim location for a different time and place, and (iii) the performance of our protocol is independent of (a) the distance between a mobile user and the location proof provider, and (b) the location tag size.

Organisation of the paper:
In Section II we present the related work. The entities and the trust and threat model of the system are explained in Section III. We present the protocol design in Section IV, and the protocol and security evaluation in Section V. Section VI concludes the paper.

II. Related Work

In recent years, a number of techniques have been proposed for location proof based on GPS [7], long range navigation (Loran) [17], Wi-Fi [6], [13], [22], [24] and Bluetooth [28], [33]. In [7], the authors generate a location signature from the microwave signals transmitted by GPS. The authors claim that it is impossible to forge the signature and its derived location. However, the approach is only suitable for outdoor environments and the work does not discuss about location verification procedure. In another work [17], the researchers have used Loran signals to generate location tag that is compared with a database of already generated tags from surveyed locations. The tag is used to block or allow specific applications on a mobile device whenever a person enters a location. However, Loran is not available on commercially available devices and is time invariant. Hence, it is feasible for an adversary to reproduce it.

Few state-of-the-art mechanisms turned their attention to smaller wireless networks i.e., wireless LAN to cover indoor environments. In [6], the authors have proposed a protocol that allows a mobile device to prove its presence to a Verifier with the help of an AP. The AP i.e., the Location Manager measures the round trip latency of request-reply protocol and

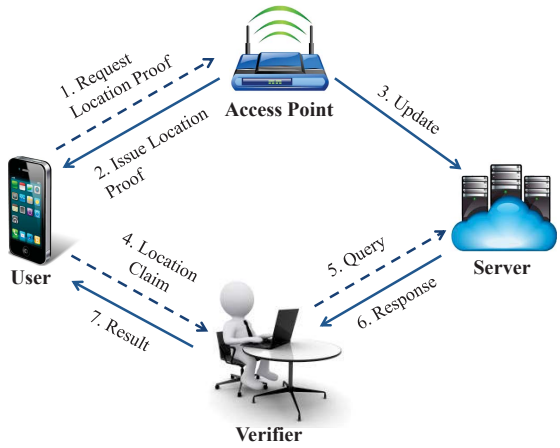
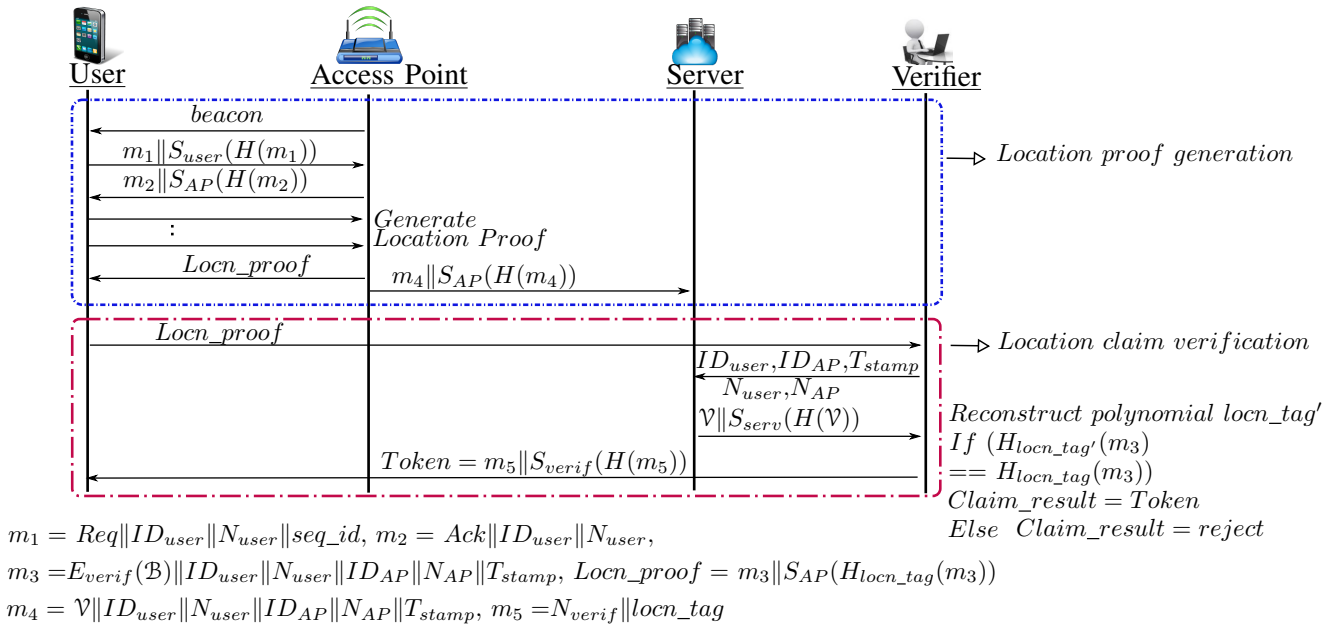


Fig. 1: System Model.

based on the time taken for the device to respond, it determines the location. Echo protocol [24] is also based on [6] which employs multiple transmitters and each of the transmitter must measure the round trip time with a specified precision. The authors in [22] have proposed a protocol for issuing a location proof by the AP to a mobile user. However, the work neither specifies the details of the approach of location proof generation and verification, nor any experimental results of how secure or robust their protocol is against any attacks. In [13], the researchers have proposed VeriPlace, a location proof architecture, however, it employs three different trusted third parties to provide location proof to the Users which is an expensive solution.

In [28] the authors have proposed a location proof mechanism for two co-located devices, a prover, i.e., mobile device and a witness who provides the location proof by employing distance bounding protocol [20]. The time required to generate location proof in this scheme linearly increases with the key size and distance between the prover and witness. Few authors have presented mechanisms for secure localisation and verification for sensor nodes that are based on distance bounding [12], [26] and time of flight of the signal [27].

The main goal of our proposed protocol is different from all the above work, specifically, our aim is to generate robust location proof based on the unique wireless channel characteristics and securely verify the same to check the legitimacy of the users in LBS and also to validate the approximate path traversed by the user. Our solution leverages existing Wi-Fi infrastructure, is not dependent on any additional entities, and does not employ distance bounding protocol which requires significant changes to the hardware for proof generation. The performance of our protocol i.e., the time required to generate the location tag remains the same irrespective of the size of location tag embedded in the location proof, and the distance between the user and AP, which is the most significant advantage over prior work and thus makes our solution suitable for practical deployment.



$m_1 = Req \| ID_{user} \| N_{user} \| seq\_id$, $m_2 = Ack \| ID_{user} \| N_{user}$,
$m_3 = E_{verif}(\mathcal{B}) \| ID_{user} \| N_{user} \| ID_{AP} \| N_{AP} \| T_{stamp}$, $Locn\_proof = m_3 \| S_{AP}(H_{locn\_tag}(m_3))$
$m_4 = \mathcal{V} \| ID_{user} \| N_{user} \| ID_{AP} \| N_{AP} \| T_{stamp}$, $m_5 = N_{verif} \| locn\_tag$

Fig. 2: Message flow between the four entities of our proposed solution.

III. System Model

A. Entities

Our system model consists of four entities as shown in Figure 1 namely:

- User: is the person who is required to prove his/her location with a location proof to LBS.
- Access Point (AP): issues a location proof to the User upon request.
- Verifier: is the one who confirms the legitimacy of the User's location claim and provides location based services.
- Server: stores the information required to verify the location claim submitted by the User. As essentially it is a database, the server can be either co-located with Verifier or a different entity.

B. Trust and Threat Model

We assume that the AP and Verifier are honest and trusted as these are the two entities which issue and verify the location proof respectively. Users are registered with LBS providers. The LBS maintains a list of trusted APs from which it can accept the proof. The User and the AP are recognised by their identities which are the public keys, and the public-private key pairs are certified by a Certificate Authority (CA). We assume that the private keys are not shared. Each public-private key pair is unique and users have them stored on their personal devices. As the devices hold sensitive personal information, people are not keen to give away their devices to others [13], hence we assume that Users do not share their personal devices.

For the threat model we consider a User who may submit a false location claim for a different place and/or time though he might not be truly present at or visited the location, by either manipulating the old location proof issued by the AP or by colluding with other users. We also assume the presence of passive eavesdropper who can eavesdrop all the communications between the entities. The eavesdropper may be present in the vicinity of the legitimate User, analyse the communications between the AP and the User, and may also follow the User's path. The passive eavesdropper may capture the CSI and generate his/her own location proof and submit to the Verifier. The adversary may also have access to the content of the Server, who may try to modify the information related to location proof.

IV. Protocol Design

Our proposed protocol consists of two phases: location proof generation and location claim verification. The sequence of communication between the four entities of the protocol is shown in Figure 2. The notation $S_{priv}(msg)$ denotes the signature of message msg with private key $priv$ and $H(msg)$ is the hash operation on msg. $E_{pub}(msg)$ and $H_z(msg)$ represent the encryption and secure hash operation on msg using the public key pub and secret key z respectively. The symbol $\|$ represents the concatenation. SHA-1 was employed for the hash function. The following subsections explain the two phases of the protocol in detail.

A. Location Proof Generation

A typical indoor wireless infrastructure consists of at least one AP. These APs advertise their capability of providing location proof through the periodically transmitted beacons consisting of the sequence identifier, seq_id. The User has to only scan for the available APs and identify them. User sends a request $m_1 \| S_{user}(H(m_1))$ where m_1 consists of the request – Req, ID of the user – ID_{user}, a nonce – N_{user} and the seq_id of the beacon. The AP checks that the seq_id in

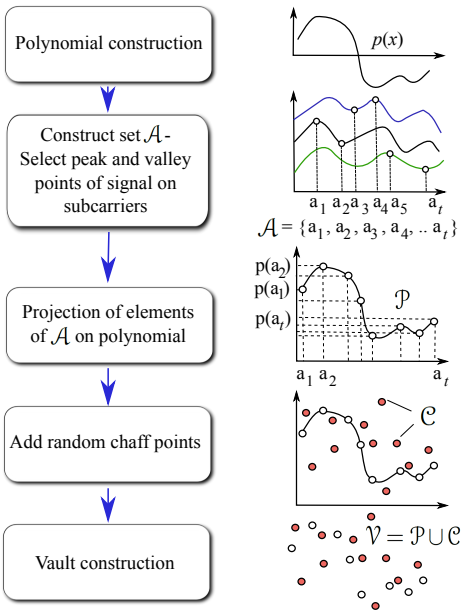


Fig. 3: Vault construction by the AP.

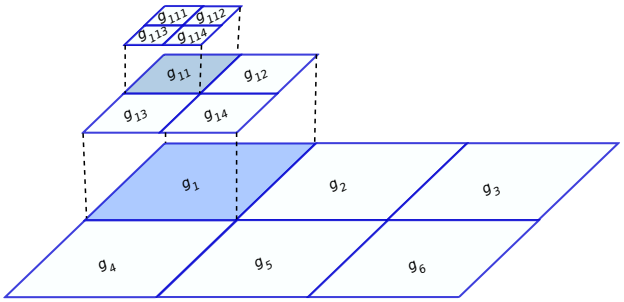


Fig. 4: Logical division of an area covered by an AP into grids and micro-grids.

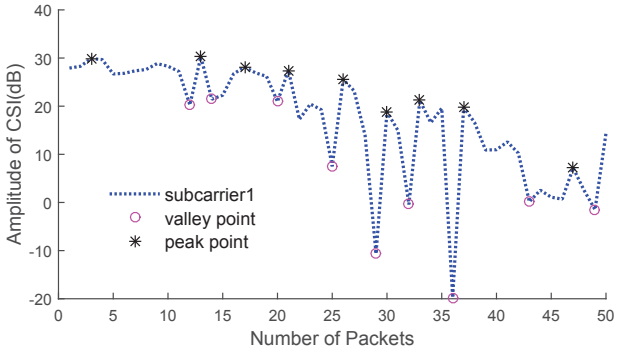


Fig. 5: Valid peak and valley points selected by AP in a window size $\mathcal{W} = 50$ for a subcarrier.

the request received is the one it has transmitted most recently, within a short specified time e.g., 100 msec, verifies the signature of the User and in turn sends an acknowledgement, $m_2 \parallel S_{AP}(H(m_2))$, where $m_2 = Ack \parallel ID_{user} \parallel N_{user}$ to the User. The User then starts sending periodic packets to the AP. Alternatively, regular scan/ping packets transmitted by the mobile devices can also be used for this purpose. The AP extracts the CSI from all the received packets. Following are the steps to generate the location proof by the AP for the User and are presented in Figure 3.

(i) *Constructing the polynomial:* We compute the effective CSI, CSI_{eff} of each packet from all the 30 subcarriers as described in [29], and estimate the approximate distance of the User. The approximate distance from the AP is evaluated by considering the mean of CSI_{eff} in a block size of 10 packets. Note that our aim is not to determine the precise location of the user, rather generate a location proof for a user in an indoor environment which proves his/her proximity in the region of interest. Hence, we focus mainly on the location proof generation and verification procedure and do not describe in detail about the fine grained localisation which has been thoroughly studied by many researchers [5], [29]. Now we need to map the mean CSI_{eff} values to the indoor location environment in which the User is/was present. We logically divide the space covered by an AP into grids. We have represented the grids in the form of square grids as shown in Figure 4. For illustration, we have shown a surface with only 6 grids. Each grid is associated with a unique grid identifier (id). Each grid can be further divided to micro-grids of smaller size. The AP consists of a database of mapping the grid id with CSI_{eff} calculated during off-line phase. The computed mean of CSI_{eff} is matched to the corresponding grid during online phase. The AP decides the granularity level of the grid. The concatenation of all the grid identifiers forms the $locn\_tag$. The elements of the $locn\_tag$ are used to construct a polynomial $p(x)$ of degree k, i.e., the elements of $locn\_tag$ form the coefficients of the polynomial.

(ii) *Construction of set $\mathcal{A}$:* The AP randomly picks a subcarrier and within a specific window $\mathcal{W}$ detects 'valid' peak and valley points which is carried out in two phases: first, all the peak and valley points of the signal within $\mathcal{W}$ are identified, and next, a threshold is used to select a subset of peak and valley points which are sufficiently apart. A peak/valley value is considered as a valid point if the difference between itself and its predecessor is greater than a threshold value. It must be noted that the number of valid peak points may not be necessarily equal to number of valid valley points. Similarly, the number of valid peak and valley points need not be equal in each $\mathcal{W}$. Figure 5 shows the peak and valley points for one of the subcarriers of the signal for $\mathcal{W} = 50$, where the threshold between the successive consecutive points is set as 5. The corresponding distinct x co-ordinates of a subset of all peak and valley points of the signal constitute a set $\mathcal{A}$ of size t, where $t \geqslant k$. The AP then computes the polynomial projections $p(\mathcal{A})$, and thus $\mathcal{P} = \{\mathcal{A}, p(\mathcal{A})\}$.

(iii) *Adding the chaff points:* The $locn\_tag$ is the original message to be verified by the Verifier. The adversary must not be able to modify the content of this message nor be able to transfer it to any other person who can gain privileges based on location. Hence, a number of random chaff points $\mathcal{C}$ are added to $\mathcal{P}$. The chaff points are randomly selected (x_i, y_i)

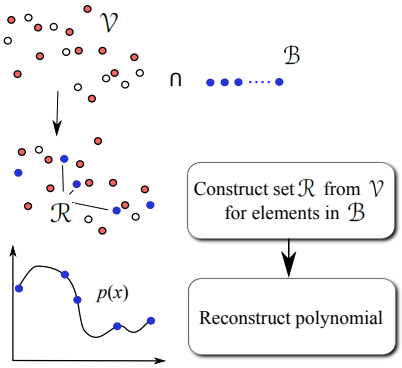


Fig. 6: Polynomial reconstruction by the Verifier.

that do not lie on $\mathcal{P}$. The chaff points must be placed in such a manner that an adversary must not be able to identify the real points by statistical analysis.

(iv) *Vault construction:* The AP now constructs the vault, $\mathcal{V} = \mathcal{P} \cup \mathcal{C}$ of size v, where $v \geq t$. The security of the vault $\mathcal{V}$ depends on the number of chaff points i.e., random noise added to $\mathcal{P}$. As the number of chaff points are increased, the security of $\mathcal{V}$ increases. The attacker must not able to distinguish between the real and chaff points without additional information provided. The HIDE Algorithm 1 presents the vault construction by AP.

(v) *Construction of set $\mathcal{B}$:* The AP constructs a set $\mathcal{B}$ whose elements substantially overlap with the elements of $\mathcal{A}$. The number of elements of $\mathcal{A}$ is equal to the number of elements of $\mathcal{B}$ i.e., the size of $\mathcal{B}$ is also equal to t. $\mathcal{B}$ is constructed by AP in such a manner that $\mathcal{A} \cap \mathcal{B}$ is at least $\geq (k+1)$. The location proof is provided to the User, where $Locn\_proof = m_3 \parallel S_{AP}(H_{locn\_tag}(m_3))$ and $m_3 = E_{verif}(\mathcal{B}) \parallel ID_{user} \parallel N_{user} \parallel ID_{AP} \parallel N_{AP} \parallel T_{stamp}$. T_{stamp} is the time at which the AP generated the location proof. The AP updates the database of the Server with the data $m_4 \parallel S_{AP}(H(m_4))$ where the term $m_4 = \mathcal{V} \parallel ID_{user} \parallel N_{user} \parallel ID_{AP} \parallel N_{AP} \parallel T_{stamp}$. The User has to submit the *Locn_proof* to the Verifier when he/she intends to the claim location. The following subsection explains the location claim verification by the Verifier.

B. Location Claim Verification

The Verifier upon receiving *Locn_proof* from the User, extracts the IDs, T_{stamp} and nonces from m_3 and submits to the Server. The Server retrieves the corresponding $\mathcal{V}$ and forwards it to the Verifier. Now the Verifier reconstructs the polynomial as shown in Figure 6. In the usual fuzzy vault scheme [11], error correction codes are employed to reveal the secret. Our protocol employs *Lagrange interpolation* to reconstruct the location tag. The Verifier reconstructs the $locn\_tag'$ from $\mathcal{V}$ as follows: The Verifier constructs a set $\mathcal{R} = \{(x,y)|(x,y) \in \mathcal{V}, x \in \mathcal{B}\}$. The set $\mathcal{R}$ must have sufficient number of points matching with the points on the polynomial i.e., if the Verifier is aware of at least $(k+1)$ points, then the vault can be revealed. *Lagrange interpolation* is used for reconstructing the polynomial based on the points in $\mathcal{R}$. The REVEAL Algorithm 2 illustrates the steps to reconstruct the polynomial for location claim verification by the Verifier.

Once the Verifier reconstructs the $locn\_tag'$, it computes $H_{locn\_tag'}(m_3)$. If the value computed is equal to the one submitted by the User then $locn\_tag' = locn\_tag$ and the Verifier sends a token that is associated with the nonce of the Verifier and the $locn\_tag$, else the location claim is rejected.

C. Numerical Example

In this subsection, we present a numerical example for illustrating the protocol. If $locn\_tag = \{1,2\}$ then the AP constructs a polynomial $p(x) = x+2$, such that the elements of $locn\_tag$ are coefficients of the polynomial. Let $\mathcal{A} = \{1,4,5\}$ and the projection of set $\mathcal{A}$ on p is $\mathcal{P} = \{(1,3),(4,6),(5,7)\}$. In order to hide $locn\_tag$, random chaff points $\mathcal{C}$ are added to the projections. Let $\mathcal{C} = \{(0,1),(3,6),(8,4)\}$, then $\mathcal{V} = \mathcal{P} \cup \mathcal{C} = \{(0,1),(1,3),(3,6),(4,6),(5,7),(8,4)\}$. Now, in order to learn the value of $locn\_tag$, the Verifier must be able to decode at least $(k+1)$ points in $\mathcal{V}$ which are correct. Let $\mathcal{B} = \{1,5,2\}$, then Verifier constructs a set $\mathcal{R} = \{(1,3),(5,7)\}$ which has at least 2 points matching with the points on the polynomial. Further, the Verifier employs *Lagrange interpolation* to reconstruct the secret $locn\_tag$. The security of the algorithm is based on the reconstruction of the polynomial. The set of chaff points $\mathcal{C}$ conceal the p from an adversary and the security increases by increasing the number of chaff points in $\mathcal{V}$.

D. Algorithm: HIDE and REVEAL

In this section, we present the HIDE and REVEAL definition of our protocol. We consider a finite field $\mathcal{F}$ of cardinality q and the sets $\mathcal{A}, \mathcal{B} \in \mathcal{F}^q$. The HIDE algorithm locks the input, a secret $locn\_tag$ with the set $\mathcal{A}$ and outputs a vault $\mathcal{V}$. The parameters are selected such that $k \leq t \leq v \leq q$.

Algorithm 1 HIDE

Input: set $\mathcal{A}$= $\{a_i\}_{i=1}^{t}$; $locn\_tag$
Output: Vault $\mathcal{V} = x_i, y_i \in \mathcal{F}$
Begin
  $\mathcal{V}, \mathcal{P} \leftarrow \varnothing$
  $p \leftarrow$ construct polynomial from elements in $locn\_tag$
  for $i = 1$ to t **do**
    $\mathcal{P} = \mathcal{P} \cup (a_i, p(a_i))$
  end for
  $\mathcal{V} = \mathcal{P}$
  for $i = (t+1)$ to v **do**
    $x_i = x_i \in \mathcal{F}^q - \mathcal{P}\{x_i\}$
    $y_i = y_i \in \mathcal{F}^q - \mathcal{P}\{y_i\}$
    $\mathcal{V} = \mathcal{V} \cup (x_i, y_i)$
  end for
  Output $\mathcal{V}$
End

The REVEAL algorithm takes the vault $\mathcal{V}$ as input and unlocks it using the set $\mathcal{B}$ to output $locn\_tag'$. If sets $\mathcal{A}$ and $\mathcal{B}$

sufficiently overlap then $locn\_tag = locn\_tag'$ else the output of the REVEAL algorithm is *null*. Appendix A presents the proposition of completeness of the two algorithms.

Algorithm 2 REVEAL

Input: Vault $\mathcal{V}$, set $\mathcal{B} = \{b_i\}_{i=1}^{t} \in \mathcal{F}$
Output: $locn\_tag'$
Begin
  $\mathcal{R}, locn\_tag' \leftarrow \varnothing$
  for $i = 1$ to t **do**
    $\mathcal{R} \leftarrow$ extract (x_i, y_i) from $\mathcal{V}$ such that x_i exists in $\mathcal{B}$
  end for
  Reconstruct $locn\_tag'$ based on $\mathcal{R}$ using *Lagrange interpolation*
  Output $locn\_tag'$
End

V. Evaluation

We have conducted an extensive set of experiments to evaluate the performance of our protocol. The experiments were carried out in three different environments, viz., (i) a large room with multiple cubicles, (ii) a medium-sized room with multiple cubicles, and (iii) a long corridor. Figure 7 shows the floor plans of the three environments and set-up used for the experiments. Laptops equipped with Intel 5300 NIC [9] were used as APs to measure CSI and android smartphones were used as hand held devices[1]. Similar to commercially available APs, we set the beacon interval time to 100 msec. The mobile devices scan for APs at regular intervals, which is a configurable parameter and was set to 2 sec. The experiments were conducted for two users, one male and one female separately who were holding the smartphone in his/her hand and followed the paths as shown in Figure 7 in each of the set-up 20 times walking at a speed of 1 m/sec. All the environments had at least 3 to 4 people walking around and performing daily routine activities. We used 3 APs to verify the consistency of our protocol. Each experiment was conducted for 5–10 minutes. Additionally, the experiments were conducted for low activity scenarios in which the user was sitting at a cubicle or on a chair with smartphone in his/her hand/pocket and occasionally getting up and walking in the room. The following subsections explain the experimental results and security analysis.

A. Results

1) Complexity of regenerating the location tag: The hardness of the fuzzy vault scheme is based on the polynomial reconstruction problem [11] which are discussed below.

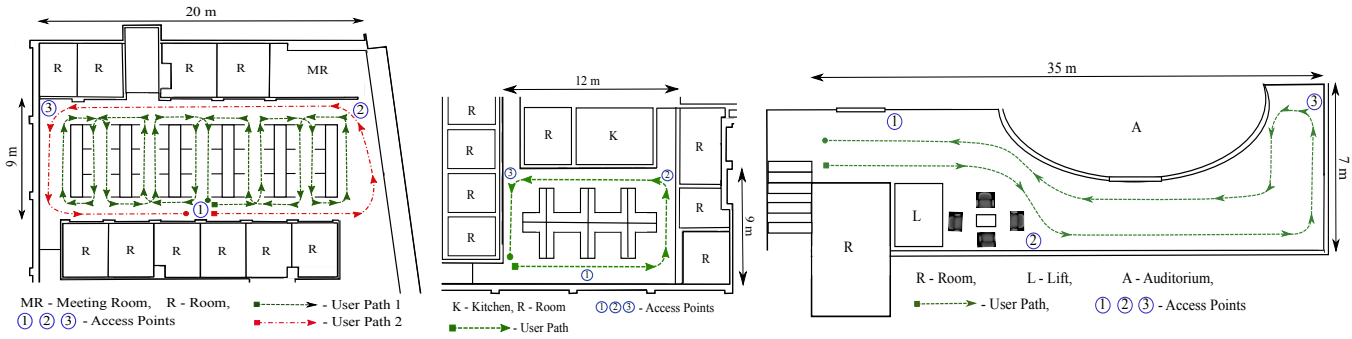
i. If the degree of polynomial is known: In this case k must be communicated to the Verifier by the AP in order to generate a fixed length $locn\_tag'$ e.g., 128 or 256 bits. Verifier can pick any $k+1$ points of the intersected data $\mathcal{R}$ and reconstruct the polynomial and generate the $locn\_tag'$. The complexity of regenerating the $locn\_tag'$ is $O(1)$.
ii. If the degree of polynomial is unknown: The Verifier initially constructs the polynomial from all the intersected points obtained. This polynomial order may be greater than the polynomial order k and has the correct coefficients embedded beginning from the lowest order of the polynomial i.e., c_0. The Verifier picks the coefficient from the lowest order of the polynomial and computes hash on m_3, which is checked for equality with the submitted value from the User. If the computation result does not match then the Verifier considers two coefficients i.e., $c_1x + c_0$. The Verifier repeats the process till the hash value is matched. Hence the complexity of the operation is $O(n+1)$, where n is the number of operations. The following Algorithm 3 which is the modified version of Algorithm 2 illustrates the procedure of the Verifier when the degree of the polynomial is unknown.

Algorithm 3 REVEAL_MOD

Input: Vault $\mathcal{V}$, set $\mathcal{B} = \{b_i\}_{i=1}^{t} \in \mathcal{F}$, m_3
Output: $locn\_tag'$
Begin
  $\mathcal{R} \leftarrow \varnothing$, $locn\_tag' \leftarrow \varnothing$
  for $i = 1$ to t **do**
    $\mathcal{R} \leftarrow$ extract (x_i, y_i) from $\mathcal{V}$ such that x_i exists in $\mathcal{B}$
    $locn\_tag' \leftarrow$ Reconstruct polynomial based on $\mathcal{R}$ using *Lagrange interpolation*
    Compute $H_{locn\_tag'}(m_3)$
    if $H_{locn\_tag}(m_3)$ == $H_{locn\_tag'}(m_3)$ **then**
      break
    end if
  end for
  Output $locn\_tag'$
End

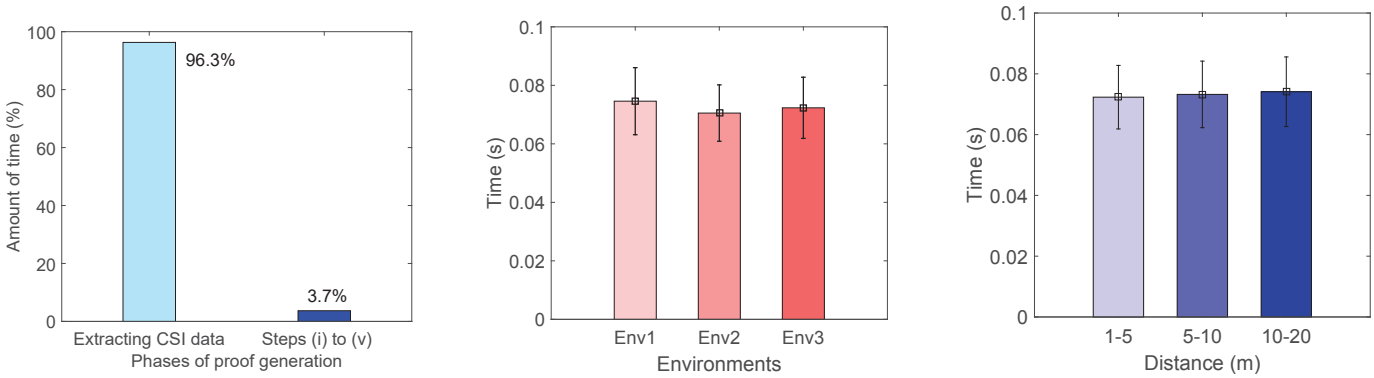
2) Performance evaluation: In this section, we evaluate the amount of time required for our protocol to generate location proof in different environments. Here we focus on the AP's part of proof generation. The processing of CSI by the AP is undertaken once the reception of the packets is completed. The various stages of location proof generation protocol i.e., steps (i) – (v) consume only $3.7\%(\approx 4\%)$ of the total time which is extremely less compared to the CSI data processing i.e., $96.3\%(\approx 96\%)$ of total time as shown in Figure 8a. From the experimental results we observed that the amount of time required to generate location tag of any size e.g., 128/256 bits was nearly same (i.e., steps (i) – (v)) of location proof generation. Figure 8b shows that the evaluation time of the protocol in all the three different environments (illustrated in Figure 7) is similar. We also evaluated the time taken by our protocol by varying the distance between the User and AP.

[1]Current off-the-shelf devices may not have the capability to measure CSI. However, we believe that in future this support may become ubiquitous. For e.g., another chipset from Atheros [1] supports CSI



(a) Environment 1 (b) Environment 2 (c) Environment 3

Fig. 7: Floor plan for experimental set-up in three different environments.



(a) Amount of time taken by different phases of location proof generation

(b) Evaluation time in three different test environments

(c) Time taken by the protocol for various distances between the User and AP

Fig. 8: Performance of the protocol to generate location proof.

As observed from Figure 8c, the amount of time required to generate the location proof is independent of the distance between the User and AP, whereas in existing scheme [28], the performance degrades with distance i.e., the protocol takes much longer time when the distance between the prover and location proof provider increases. Compared to the state-of-the-art (3.4 s) [28], our protocol is faster by an order of magnitude.

B. Security Evaluation

1) Attack complexity of $\mathcal{V}$: In this section, we discuss the computational complexity of an adversary who tries to predict the *locn_tag* by analysing the $\mathcal{V}$ stored in the Server. Security of the vault $\mathcal{V}$ depends on the size of $\mathcal{C}$ i.e., noise added to $\mathcal{P}$. In our proposed protocol, in order to determine the security of $\mathcal{V}$, we evaluate the number of computations the adversary has to perform in order to unlock the vault. The adversary has to try out all possible combinations of each $(k+1)$ points in $\mathcal{V}$ to obtain the legitimate points. Here an adversary is brute-forcing to derive the correct polynomial. Mathematically, if $v = 400$, $k = 14$, an adversary has to try a total combinations of C(400, 15) $\approx$ 6.29E+26 computations which is equivalent to predicting a key of 89-bit security. Figure 9 shows that as the number of chaff points are increased in the vault the complexity of the adversary also increases. Depending on the security level required, more number of random points can be added to the projections. For most applications 85-bit security offers good cryptographic security [11], [15].

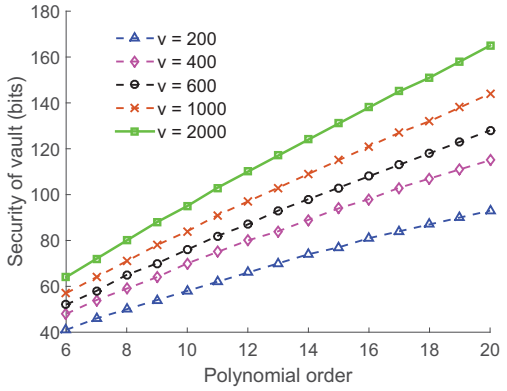


Fig. 9: Security of the vault increases with increase in polynomial order and vault size.

2) Security properties: In this section, we present and analyse the security properties of our protocol.

Property 1: A User cannot create his/her own location claim and submit it to the Verifier.

The data related to the location proofs submitted by the AP are

updated in the Server. Hence, even if the User claims a fake location, it can be detected by the Verifier as the corresponding information is not present in the database.

Property 2: A User cannot use the location claim to gain benefits for a different location and time.

The $Locn\_proof$ consists of the $locn\_tag$, the *IDs* of the User and AP and the time at which the proof was generated. Hence, the proof is unique to a location and time. If a User attempts to utilise the same location proof for a different location, the corresponding AP will have a different ID and the Verifier will be able to detect the false location claim. A user may also try to claim location at the same place but at a different time, which is not possible because of the T_{stamp} embedded in the proof. Another security factor is that, as the AP updates the details of every location proof provided to the User in the Server, the Verifier will not be able to obtain the corresponding information related to the fake location proof submitted.

Property 3: The location claim cannot be transferred or shared by a User with another User.

If a dishonest user say User1 transfers his/her location proof to User2, and User2 embeds his/her ID instead of User1 to claim benefits, then the false location claim can be immediately identified by the Verifier as the Sever will not have any entries corresponding to the ID_{user} and T_{stamp}. Also, changing the ID_{user} produces a different hash value which is not equal to the original value provided by the AP, and by the properties of hash we know that it is infeasible to find another input that gives the same hash value. Hence the location tag cannot be transferred to another User. The properties of hash function are presented in Appendix B.

Property 4: It is infeasible for an adversary to obtain $locn\_tag$ from vault $\mathcal{V}$.

The $locn\_tag$ is hidden in the vault $\mathcal{V}$ by the chaff points. The security of the vault depends on the number of chaff points addded. From Figure 9, we observe that for $v = 600$, when k is increased from 5 to 20, the security level increases from 50 to 125 bits. For a good cryptographic application, the security level of at most 85-bit is sufficient [11], [15].

Property 5: An adversary cannot modify the $Locn\_proof$ and the content of the Server so as to deny the LBS benefits to a legitimate User.

Our protocol ensures that the data actually originated from the source and the state is unmodified. The $Locn\_proof$ provided by the AP to the User consists of m_3 and the hash value of the message with the $locn\_tag$. The set $\mathcal{B}$ cannot be modified by the User nor any adversary without any additional information. Neither can the adversary modify any part in the Server i.e., $\mathcal{V}$, IDs, etc., as the hash of the message is also stored.

Property 6: An adversary cannot modify/access the token provided by the Verifier to the User.

The token provided to User is signed by the Verifier and acts as a key to access the services. This signature is verified by the User to ensure that the message has originated from the Verfier and the hash reveals that the token has not been modified.

3) Passive eavesdropper: The unique properties of wireless physical layer characteristics has been extensively studied for authentication [30] and key generation [14], [21], [31]. Researchers have analysed and illustrated that an eavesdropper who is present in the vicinity of the legitimate devices at a distance greater than half the wavelength of the radio frequency used observes entirely different wireless characteristics than the two legitimate communicating devices. This is due to the multipath fading effects [18]. Hence, a passive eavesdropper who overhears all the communication between an AP and User, cannot generate the same location tag as the legitimate ones and claim location benefits. In our scheme the communication between all the entities is secured by TLS/SSL.

4) Discussion: Some of the possible threats like relay attack where a User1 relays the communication with AP to User2 can be overcome by incorporating second factor authentication schemes by using biometric sensors, cameras [8], microphone of the mobile devices [16] and radio fingerprinting [19]. To make attacks infeasible and identifiable, a very small time limit can be set in the round trip communication between the AP and the User, so that if the User fails to respond within a specified time, the request is rejected [23].

VI. Conclusion

In this paper we have proposed a novel secure location proof generation and verification protocol for mobile users and LBS by employing information theoretically secure fuzzy vault scheme and unique spatial-temporal wireless channel characteristics. Our theoretical and experimental results prove that our protocol offers good cryptographic security. Our protocol uses existing Wi-Fi infrastructure and can be implemented by employing commercially available off-the-shelf devices. We also show experimentally that, compared to the state-of-the-art, (i) the time required to generate the location tag in our solution is independent of the size of location tag generated and the distance between the User and AP, and (ii) our scheme is faster by an order of magnitude. In our future work we would like to consider the cases where APs may collude with adversaries to generate a fake location proof and services that may deny access to honest users or grant illegitimate access to dishonest users.

Acknowledgment

This work is partially supported by Australian Research Council Discovery grant DP150100564.

References

[1] "Atheros chipset," http://pdcc.ntu.edu.sg/wands/Atheros/, accessed: 6-May-2016.
[2] "Foursquare," https://foursquare.com/, accessed: 6-May-2016.
[3] "Yelp," http://www.yelp.com/, accessed: 6-May-2016.
[4] "IEEE Std. 802.11n-2009: Enhancements for higher throughput," http://www.ieee802.org, 2009.
[5] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based User Location and Tracking System," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2000.
[6] W. Brent and F. Edward, "Secure, Private Proofs of Location," Princeton University, Tech. Rep., 2003, https://www.cs.princeton.edu/research/techreps/TR-667-03.
[7] D. E. Denning and P. F. MacDoran, "Location-based Authentication: Grounding Cyberspace for Better Security," in *Computer Fraud and Security*, Feb. 1996.

[8] P. Gilbert, J. Jung, K. Lee, H. Qin, D. Sharkey, A. Sheth, and L. P. Cox, "YouProve: Authenticity and Fidelity in Mobile Sensing," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2011.
[9] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 Packet Delivery from Wireless Channel Measurements," in *Proceedings of the ACM SIGCOMM Conference*, 2010.
[10] C. Javali, G. Revadigar, W. Hu, and S. Jha, "Poster: Were You in the Cafe Yesterday?: Location Proof Generation & Verification for Mobile Users," in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2015.
[11] A. Juels and M. Sudan, "A Fuzzy Vault Scheme," *Des. Codes Cryptography*, vol. 38, no. 2, pp. 237–257, 2006.
[12] L. Lazos, R. Poovendran, and S. Čapkun, "ROPE: Robust Position Estimation in Wireless Sensor Networks," in *Proceedings of the International Symposium on Information Processing in Sensor Networks (IPSN)*, 2005.
[13] W. Luo and U. Hengartner, "VeriPlace: A Privacy-Aware Location Proof Architecture," in *Proceedings of the ACM International Symposium on Advances in Geographic Information Systems (ACM-GIS)*, 2010.
[14] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik, "Radio-telepathy: Extracting a Secret Key from an Unauthenticated Wireless Channel," in *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2008.
[15] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, 1996.
[16] M. Miettinen, N. Asokan, F. Koushanfar, T. D. Nguyen, J. Rios, A. Sadeghi, M. Sobhani, and S. Yellapantula, "I Know Where You are: Proofs of Presence Resilient to Malicious Provers," in *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, 2015.
[17] D. Qiu, S. Lo, P. Enge, and D. Boneh, "Geo-encryption using Loran," in *Proceedings of the National Technical Meeting of the Institute of Navigation*, 2007.
[18] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Prentice Hall, 2001.
[19] K. B. Rasmussen and S. Capkun, "Implications of Radio Fingerprinting on the Security of Sensor Networks," in *Proceedings of the International Conference on Security and Privacy in Communication Networks and the Workshops (SecureComm)*, 2007.
[20] K. B. Rasmussen and S. Čapkun, "Realization of RF Distance Bounding," in *Proceedings of the USENIX Conference on Security*, 2010.
[21] G. Revadigar, C. Javali, W. Hu, and S. Jha, "DLINK: Dual Link Based Radio Frequency Fingerprinting for Wearable Devices," in *Proceedings of the IEEE Conference on Local Computer Networks (LCN)*, 2015.
[22] S. Saroiu and A. Wolman, "Enabling New Mobile Applications with Location Proofs," in *Proceedings of the Workshop on Mobile Computing Systems & Applications (HotMobile)*, 2009.
[23] ——, "I Am a Sensor, and I Approve This Message," in *Proceedings of the Workshop on Mobile Computing Systems & Applications (HotMobile)*, 2010.
[24] N. Sastry, U. Shankar, and D. Wagner, "Secure Verification of Location Claims," in *Proceedings of the ACM Workshop on Wireless Security (Wise)*, 2003.
[25] U. Uludag, S. Pankanti, and A. K. Jain, "Fuzzy Vault for Fingerprints," in *Audio- and Video-Based Biometric Person Authentication*, 2005.
[26] S. Čapkun and J. Hubaux, "Secure Positioning of Wireless Devices with Application to Sensor Networks," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2005.
[27] S. Čapkun, K. Rasmussen, M. Čagalj, and M. Srivastava, "Secure Location Verification with Hidden and Mobile Base Stations," *IEEE Transactions on Mobile Computing*, vol. 7, no. 4, pp. 470–483, 2008.
[28] X. O. Wang, J. Zhu, A. Pande, A. Raghuramu, P. Mohapatra, T. F. Abdelzaher, and R. K. Ganti, "STAMP: Ad Hoc Spatial-Temporal Provenance Assurance for Mobile Users," in *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, 2013.
[29] K. Wu, J. Xiao, Y. Yi, M. Gao, and L. M. Ni, "FILA: Fine-grained Indoor Localization," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2012.
[30] L. Xiao, L. J. Greenstein, N. B. Mandayam, and W. Trappe, "Fingerprints in the Ether: Using the Physical Layer for Wireless Authentication," in *Proceedings of the IEEE International Conference on Communications (ICC)*, 2007.
[31] C. Ye, S. Mathur, A. Reznik, Y. Shah, W. Trappe, and N. B. Mandayam, "Information-theoretically Secret Key Generation for Fading Wireless Channels," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 240–254, 2010.
[32] Z. Zhang, L. Zhou, X. Zhao, G. Wang, Y. Su, M. Metzger, H. Zheng, and B. Y. Zhao, "On the Validity of Geosocial Mobility Traces," in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets)*, 2013.
[33] Z. Zhu and G. Cao, "APPLAUS: A Privacy-Preserving Location Proof Updating System for Location-based Services," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2011.

APPENDIX

A. Completeness of HIDE and REVEAL algorithms

The following proposition provides the completeness of the two algorithms HIDE and REVEAL. We assume that the probability is overwhelming if it is larger than $1-\omega$, where ω is a negligible quantity and the probability is negligible if it is asymptotically smaller than any positive polynomial in t [11].

Proposition 1: A $(\mathrm{HIDE}, \mathrm{REVEAL})$ pair with the input parameters (k, t, v, q) such that $k \leq t \leq v \leq q$ is complete with Δ-fuzziness if and only if the following satisfies: For every $\mathcal{A}, \mathcal{B} \in \mathcal{F}^q$, such that $|\mathcal{A} - \mathcal{B}| \leq \Delta$, $\mathrm{REVEAL}(\mathcal{B}, \mathrm{HIDE}(\mathcal{A}, locn\_tag)) = locn\_tag$ with overwhelming probability.

B. Properties of hash function

A hash function with inputs x, x' and outputs y, y' has the following properties:

i. preimage resistance - it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x' such that $h(x') = y$ when given any y for which a corresponding input is not known.
ii. 2nd preimage resistance - it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a 2nd-preimage $x' \neq x$ such that $h(x) = h(x')$.
iii. collision resistance - it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output, i.e., such that $h(x) = h(x')$.