Enabling Collusion Resistant Location Proof And Secure Location Sharing For Mobile Users

C. Subhashri and Dr. P. R. Vijayalakshmi

1PG Student, Department of Computer Science and Engineering, K.L.N. College of Engineering, Affiliated to Anna University, Chennai.
2Professor, Department of Computer Science and Engineering, K.L.N. College of Engineering, Affiliated to Anna University, Chennai.

Received 28 January 2017; Accepted 22 March 2017; Available online 4 April 2017

Address For Correspondence:
C.Subhashri, PG Student, Department of Computer Science and Engineering, KLNCE, Pottapalayam, Sivangangai – 630 612, Tamil Nadu, India.
E-mail: csmrp91@gmail.com

Copyright © 2017 by authors and American-Eurasian Network for Scientific Information (AENSI Publication).
This work is licensed under the Creative Commons Attribution International License (CC BY).
http://creativecommons.org/licenses/by/4.0/

ABSTRACT
Location based services are immensely popular. Many location based services depends on current location. But determine the past location is important. Malicious users may access the restricted resources by cheating on their locations. Spatial Temporal provenance Assurance with Mutual Proof (STAMP) scheme used to find the past location of the users effectively. At the time of finding location proof there is a chance to occur collusion attack between the users and users may try to change their identity to get the location proof. So Collusion Resistance is one of the important aspects in the STAMP scheme. In order to defend against colluding attacks, present the Ciphertext Policy Attribute-Based Encryption (CP-ABE) method. Location sharing is the method to share the user’s location in secure manner. To provide secure sharing of location proof Blowfish algorithm is used. Extensive experimental results show that CP-ABE can effectively defend against colluding attacks and Blowfish algorithm can effectively share the location proof in secure manner.

KEYWORDS: Location based Services, location proof, location privacy, colluding attacks, location sharing

INTRODUCTION

As location based mobile device are immensely growth. Most of the technique based on the current locations. Users find their current location and send it to the server and the server provides the resources to the user based on the current locations. It is challenge to find the users geographical locations. Saroiu et al explained several applications in [7] that are(1) A general problem on auction websites as ebay is account theft-attackers break into valid accounts and use their customary reputations to commit fraud.(2)Many police investigations are quickly resolved by tentative the alibis of the persons involved in an incident. With location proofs, people can use their mobile phones to make such alibis.(3) During an election, voters are frequently asked to provide proof of their existence in particular region, state or country for a pre determined period of time.

The above applications need users to be able to obtain proofs from the locations they stopover .Users may then choose to present one or more of their proofs to a third-party verifier to claim their existence at a location at a specific time. Geo-location data is collected in many ways, including Global Positioning System devices, IP address, or Wi-Fi network mapping. Location proof plays a very important role in location based applications. Location proof is a piece of data that certifies spatial and temporal information about the mobile device of the users. In the location proof updating system, location information is able to be stolen by adversaries. It may
cause exposure towards location privacy of the user. Public key Cryptographic technique is used for encryption and decryption of communicating messages and secure from eavesdropping.

The important issue and design challenge in location proof is collusion attack. Collusion is an agreement between two or more parties, sometimes illegal and therefore secretive, to limit open competition by unreliable, misleading, or defrauding others of their legal rights. To avoid collusion attacks Ciphertext-Policy Attribute Based Encryption (CP-ABE) method is used. While all the ABE systems endure from the problem, some recent attempts have been made to accomplish traceability, exclusively, there are two levels of traceability: (1) given a well-rounded decryption key, a Whitebox tracing algorithm can used to discover out the original key owner; and (2) given a encryption-device while the fundamental decryption algorithm or key could not be given, a Blackbox tracing algorithm, which treats the decryption-device as an oracle, be capable to get out at least one of the malevolent users whose keys have been used for structure the decryption-device. While the Blackbox Traceable CP-ABE scheme in is highly important, fully secure in the customary model, and efficient in getting fully collusion-resistant blackbox traceability.

Location sharing is the method to share location proof for getting resource. The possibility of grouping users in proportion to geographical region provides location-sharing-based services (LSBS). This category of services match LBSs using information from a group of users and not just from individuals, providing contextualized services based on the location of a group. Using privacy techniques and blowfish method, the implementation of the model guarantees not only the privacy of the group but also the privacy of each one inside the group.

Related Works:

Toward Privacy Preserving and Collusion Resistance in a Location Proof Updating System was discussed by Zhichao Zhu et al [2]. They proposed A Privacy-Preserving LocAction proof Updating System (APPLAUS) Scheme for mutually generate location proofs and send updates to a location proof server. Betweenness ranking-based and correlation clustering-based approaches are used to defend against collusion. Xinlei Wang et al [1] proposed a STAMP: Enabling Privacy-Preserving Location Proofs for Mobile Users where STAMP is designed for ad-hoc mobile users generating location proofs for each other in a distributed setting. STAMP guarantees the integrity and non-transferability of the location proofs and protects users' privacy.

Wei Cheng Cheng et al [3] proposed A user sensitive privacy-preserving location sharing system where system to avoid leaking. Users define sensitivity profiles to transform the public available geographic data into personal obfuscate region maps. Shared locations from obfuscate region maps provide close enough coordinates for application to use. Model of Location-Sharing-Based Services with Privacy Guarantee was discussed by Tiago Antonio Rosa et al [4]. They proposed an implementation of a model of Location Sharing Based Services (LSBS) with privacy guarantees. This guarantee is due to homomorphic encryption and privacy techniques like anonymity.

John Bethencourt et al [5] proposed a Ciphertext-Policy Attribute-Based Encryption where a system for realizing complex access control on encrypted data that call Ciphertext-Policy Attribute-Based Encryption. By using this techniques.Encrypted data can be kept confidential even if the storage server is untrusted; moreover, these methods are secure against collusion attacks. In this system attributes are used to describe a user’s identification, and a party encrypting data finds a policy for who can decrypt. Thus, this method is conceptually closer to traditional access control methods such as Role-Based Access Control (RBAC).

CREPUSCOLO a Collusion Resistant Privacy Preserving Location Verification System was discussed by Mauro Conti et al [6]. They proposed a CREPUSCOLO, they use "location proofs" collected from co-located mobile devices, which can be endorsed by a "token" acquired from a trusted Token Provider. In fact, location-proofs endorsed by tokens offer the resiliency against collusion attacks, because this arrangement can prove that a certain mobile device was at a certain location at a specific time. CREPUSCOLO also saves the source location privacy by enforcing the handling of periodically altering each and every pseudonym.

System Model:

A. STAMP Scheme:

Wireless infrastructure may not exist everywhere and hence a system based on wireless APs creating STP proofs would not be viable for all scenarios. In addition, the deployment cost would be high if we need a large number of wireless APs to have the capability of generating STP proofs. Therefore, think a distributed STP proof architecture, i.e., mobile users getting STP proofs from close by mobile peers, would be more realistic and suitable for a wider range of applications. To design a common decentralized protocol, and then explain how it can work well for centralized case also.

Fig. 1 illustrates the architecture of the system. There are four types of entities based on their roles.

- **Prover:** A prover is a mobile device which tries to find STP proofs at a definite location.
- **Witness:** A witness is a device that is in closeness with the prover and is eager to create an STP proof for the prover ahead receiving his/her request. The witness can be untrusted or trusted.
• **Verifier**: A verifier is the party that the prover wants to find one or more STP proofs to and claims his/her survival at a location at a particular time.

• **Certificate Authority (CA)**: The CA is a semi-trusted server which produces, manages cryptographic credentials for the other parties. CA is also in charge for proof verification and trust evaluation.

A prover and a witness communicate with each other via Bluetooth or WiFi. A peer discovery mechanism for discovering nearby witness is required and preferably provided by underlying communication technology instead of this protocol. The proof generation system of prover is available a list of existing witnesses. When there are multiple witnesses ready to cooperate, the prover start protocol with them sequentially. STP claims are sent to verifiers from provers by a LAN or Internet, and verifiers are implicit to have Internet connection with CA. Each user can act as a prover or a witness, depending on their roles at the instant.

![System Architecture Diagram](image)

**Fig. 1**: An illustration of system architecture.

**B. Location Sharing**:

Location sharing framework consists of a database server and a set of (mobile) users. The database server is maintained by a location based service provider. Fig. 2 illustrates the LSBS framework, in which admin sends his/her location in encrypted form according to our Blowfish scheme to the database server. Server receives encrypted locations and provides resources to the user. Then server decrypts the location information.

![LSBS Framework Diagram](image)

**Fig. 2**: An illustration of LSBS framework.

**The Stamp Scheme**:

**A. Preliminaries**:

Spatial Temporal provenance Assurance with Mutual Proof (STAMP) scheme is used to generate the spatial Temporal Provenance (STP) Proof and used to claim the STP proof and perform verification process. STAMP aims at involving the integrity and non-transferability of the location proofs, with the capability of protecting users’ privacy. STAMP is based on a distributed architecture. Co-located mobile devices mutually generate and endorse STP proofs for each other. STAMP requires only a single semi-trusted third party which can be embedded in a Certificate Authority (CA).
B. Cryptographic methods:

A commitment scheme allows one to commit to a message while keeping it unknown to others, with the facility to expose the committed value later. The original message cannot be altered after it is committed to. One-way hash functions have the related binding and hiding properties as commitment schemes. However, for privacy protection purpose, do not apply hash functions because they are weak to dictionary attacks.

C. Protocol:

In this module to deploy the certain number of users. All users are traveling with a random mobility model. In each STP proof collection event, a random prover is selected among all the users. The user try to generate location proof from neighborhood users (Witness) in STP generation phase. Users are entering into the process to complete the registration process and then login into the phase.

Each and every node is considered as an individual user and the communication is held between those users.

**STP Proof Generation**

**Prover:** Suppose a prover wants to start an STP proof collection event at time, the prover first broadcasts an STP proof request (denoted as PReq) to other nearby mobile devices and waits for responses. A PReq is constructed as follows:

\[
PReq = C(\text{ID}_p, r_p) \mid L_1 \mid t
\]  

Here, \( \text{ID}_p \) is the prover's ID, \( r_p \) - Random nonce generated by the prover, \( L_1 \) is the lowest level of the current location and \( t \) is time period.

**Witness:** A witness who receives a PReq decides if he/she accepts the request. If the request is accepted, the witness sends an ACK back to the prover, after which, the two parties start the execution of the distance bounding stage.

The witness first creates an **STP record** (denoted as STPR):

\[
\text{STPR} = C(L_1, r_{1w}) \mid \ldots \mid C(L_n, r_{nw}) \mid t
\]  

Here, \( r_{1w} \) is the Random nonce generated by the witness and used to commit to \( L_1 \) provided in PReq. The higher location levels \( L_2, \ldots, L_n \) are also committed with different nonce \( L_2^{nw}, \ldots, L_n^{nw} \), which in turn are derived based on a hash chain operation.

**STP Claim:**

**Prover:** The prover extracts the necessary data from his/her corresponding STP proof entry and creates an STP claim (denoted as STPC).

\[
\text{STPC} = \text{EP}_1 \mid \ldots \mid \text{EP}_m \mid \text{ID}_p \mid r_p \mid L_x \mid t
\]  

Here, \( L_x \) is the lowest location level, \( r_{x,w,1}, \ldots, r_{x,w,m} \) is derived from the \( r_{1w,1}, \ldots, r_{1w,m} \) based on a hash chain operation.

**STP Verification:**

**Verifier:**

After receiving the prover’s STPC, the verifier needs CA’s assistance in verifying the STPC. The verifier now constructs a verification request (denoted as VReq) by extracting the following information from the STPC:

\[
\text{VReq} = \text{EP}_1 \mid \ldots \mid \text{EP}_m \mid \text{ID}_p \mid r_p
\]  

The VReq is then sent to the CA.

**CA:** CA creates a VRes as follows and sends it back to the verifier

\[
\text{VRes} = E_{CA}^{K_{CA}} (\text{STPR}_1 \mid \ldots \mid \text{STPR}_m \mid z)
\]  

Here, \( \text{STPR}_1 \mid \ldots \mid \text{STPR}_m \) is the STPRs extracted from \( \text{EP}_1 \mid \ldots \mid \text{EP}_m \) respectively and \( z \) is the big integer resulted from the distance bounding stage.
D. Collusion Detection:

The significant property of Attribute-Based Encryption systems are their resistance to collusion attacks. This property is critical for building cryptographic access control systems; otherwise, it is not possible to guarantee that a system will show the preferred security properties as there will survive overwhelming attacks from an attacker that manages to get a hold of a few private keys. While we might consider ABE systems with different flavors of expressibility, prior work made it clear that collusion resistance is a required property of any ABE system.

E. Protocol:

Definition for Access Structure:

Let \{Q_1, Q_2, \ldots, Q_n\} be a set of parties. A collection \( A \subseteq 2^{\{Q_1, Q_2, \ldots, Q_n\}} \) is monotone if \( \forall B, C : \text{if } B \in A \text{ and } B \subseteq C \Rightarrow C \in A. \) An access structure is a collection (respectively, monotone collection) \( A \) of non-empty subsets of \{Q_1, Q_2, \ldots, Q_n\}, i.e., \( A \subseteq \{\{Q_1, Q_2, \ldots, Q_n\}\} \).

A ciphertext-policy attribute based encryption scheme consists of five fundamental algorithms: Setup, Encrypt, KeyGen, Decrypt and Delegate.

Setup:

The setup algorithm takes as no input other than the implicit security parameter. It produces the public parameters \( PK \) and a master key \( MK \). It will choose a bilinear group \( E_0 \) of prime order \( q \) with generator \( f \). Next it will choose two random exponents \( \alpha, \beta \in \mathbb{Z}_q \). The public key is published as:

\[
PK = E_0, f, h = f\beta, \alpha s, \beta e \exists (f, f) \alpha
\]  

Encrypt \((PK, M, A)\):

The encryption algorithm takes as input the public parameters \( PK \), a message \( M \), and an access structure \( A \) above the creation of attributes. The algorithm will encrypt \( M \) and create a ciphertext \( CT \) such that only a user that possesses a set of attributes that satisfies the access structure will be able to decrypt the message.

\[
CT = (T, \forall C = M e(f, f)\alpha, C = h s, \forall Y : Cy = fqy(0), C'y = H(\text{att}(y)))^{y(0)}
\]

Key Generation \((MK, S)\):

The key generation algorithm is taking as input the master key \( MK \) and a set of attributes \( S \) which describe the key. It outputs a private key \( SK \). The algorithm first chooses a random \( r \in \mathbb{Z}_q \) and then random \( rj \in \mathbb{Z}_q \) for each attribute \( j \in S \).

\[
SK = (D = f^{a+ry}, \forall j \in S : D_j = f, H(j)^r, D_j = f_j)
\]

Delegate \((SK, S)\):

The delegate algorithm is taking as input a secret key \( SK \) for some set of attributes \( S \) and a set \( S' \subseteq S \). It output a secret key \( S'K \) for the set of attributes \( S' \).

Decrypt \((PK, CT, SK)\):

The decryption algorithm is taking as input the public parameters \( PK \), a ciphertext \( CT \), which contains an access policy \( A \), and a private key \( SK \), that is a private key for a set of attributes. If the set \( S \) of attributes satisfies the access structure \( A \) then the algorithm will decrypt the ciphertext and return a message \( M \).

\[
\text{Decrypt Node}(CT, SK, x) = \frac{e(Di, Cx)}{e(Di, C'x)}
\]

\[
E(fr, H(i)^r, h^px(0)) = (e(f^r, H(i)^px(0)) = e(f, f)rpz(0)
\]

\[= \text{C'}(e(C, D)/A) = \text{C} (e(h, f^{a+ry})/e(f, f)'s) = M\]
F. Location Sharing:
Blowfish is a symmetric block cipher which is effectively used for encryption and protecting the data. Blowfish has a 64 bit block size and a variable key length from 32 bits up to 448 bits. It can with a 16-round Feistel cipher and uses large key-dependent S-boxes. In structure it is like CAST-128, which uses fixed S-boxes. The Fig 3 shows the action of Blowfish. Each line represents 32 bits.

The algorithm keeps two sub key arrays: the 18-entry P-array and four 256 entry S-boxes. The S-boxes accept 8-bit input and gives 32-bit output. One entry of the P-array is used every round, and finish the final round, each half of the data block is XORed with one of the two remaining unused P-entries. The Fig shows Blowfish's F-function. The function separates the 32-bit input into four eight-bit quarters, and uses the quarters as input to the S boxes. The outputs are added modulo 232 and XORed to gives the final 32-bit output.

Experiments And Results:
A. Prototype implementation:
A prototype is implemented client application on NetBeans with Java. Experiments are carried out on windows 7 devices equipped with Pentium dual core, 3 GB RAM, Bluetooth, and running in NetBeans IDE 7.3.1. Bluetooth is used as the communication interface between mobile devices. Ciphertext policy Attribute Based Encryption scheme used to defend against the collusion which is identity based. Blowfish algorithm is used to effectively shared location of the users.

1) Performance:
With the implementation, examine the computational time for STAMP, time with respect to packet size for blowfish algorithm and encryption and decryption time in Ciphertext Policy Attribute Based Encryption method.

Fig. 3: The Feistel Structure of Blowfish

Fig. 4: Time to generate an STP Proof under different Key Sizes
Fig. 5: Comparison by Communication speed

Fig 4 and fig 5 explains the time taken to generate location proof and communication Speed comparison. Fig 6 and 7 describes the time to generate key generation and encryption in the Ciphertext –Policy Attribute Based Encryption method for collusion resistance.

Fig. 6: Key Generation Time

Fig. 7: Encryption Time

Table 1 describe the time with respect to the packet size. In the Blowfish algorithm there is less time to take for finish location sharing. Blowfish shows better results than AES, IDEA, DES etc…
Table I: Time consumption

<table>
<thead>
<tr>
<th>Packet Size (Kb)</th>
<th>Time (millisecond)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024.00</td>
<td>510</td>
</tr>
<tr>
<td>1500.02</td>
<td>624</td>
</tr>
<tr>
<td>2100.50</td>
<td>856</td>
</tr>
<tr>
<td>2512.67</td>
<td>975</td>
</tr>
<tr>
<td>3124.21</td>
<td>1024</td>
</tr>
</tbody>
</table>

B. Simulation:

To measure the effectiveness and accuracy of the collusion detection, implemented the trust model with Java simulation. In this section, present the simulation details and the performance results that we obtained from the simulation experiments.

1) Simulation Setup:

Since the main purpose of the simulation is to evaluate the effectiveness of the trust model in an aggressive environment, first test the case where there are no trusted mobile users. In this simulation, a total number of 1000 users are deployed. All users are traveling with a random mobility model. In each STP proof collection event, a random prover is selected among all the users. From this simulation setup find out the percentage of colluding attackers (PC), Collusion Tendency (CT), Mean of Witness (μ\_w), Standard Deviation of Witness (σ\_w), Trust Scaling Parameter (ω), Collusion trust Threshold (θ) and training location proof collection events.

Table II contains the CPABE method results based on the simulation strategy. The default simulation settings are,

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Users</td>
<td>1000</td>
</tr>
<tr>
<td>percentage of colluding attackers (PC)</td>
<td>2.23%</td>
</tr>
<tr>
<td>Collusion Tendency (CT)</td>
<td>0.4</td>
</tr>
<tr>
<td>Mean of Witness (μ_w)</td>
<td>6</td>
</tr>
<tr>
<td>Standard Deviation of Witness (σ_w)</td>
<td>3</td>
</tr>
<tr>
<td>Trust Scaling Parameter (ω)</td>
<td>1.2 x 10^-4</td>
</tr>
<tr>
<td>Collusion trust Threshold (θ)</td>
<td>0.7</td>
</tr>
<tr>
<td>Training location proof collection events</td>
<td>10000</td>
</tr>
</tbody>
</table>

2) Performance Metric:

Here, use the Balanced Accuracy (BA) as the performance metric, which is a commonly used accuracy measure for classification algorithms. BA is defined as the arithmetic mean of true positive rate and true negative rate.

3) Simulation Results:

First, two key parameters that concern how the trust model performs are the trust scaling parameter ω And θ collusion trust threshold. Run general tests to observe the BA distribution under different choices of ω and θ with the default setup. The results are shown in Fig. 6. The trust model performs well BA>0.91 only when good ω and θ are chosen.

![Fig. 8: BA under different θ and ω](image-url)
Conclusion And Future Work:

In this work, presented STAMP, that aims at providing security and privacy guarantee to mobile users' proofs for their past location visits. STAMP relies on mobile devices to mutually create location proofs. Integrity and non-transferability of location proofs and location privacy of users are the main design goals of STAMP. At the time of creating location proof there is a chance to occur collusion attack. Ciphertext-Policy Attribute Based Encryption (CPABE) is effectively used to defend against Collusion attack. The main property of CPABE is identity based which can effectively find out collusion. Location Sharing is the method to share user’s location to the server for getting resources. At the time of location sharing there is a chance to involve intruders and chance the legitimate location proof. Blowfish algorithm is advanced method for protect the user's location to the server for getting resources. At the time of location sharing there is a chance to involve intruders effectively. Experimental results shows that protocol effectively find past location visits, find out collusion attacks and sharing the location in secure manner. Implementation with java indicates low computational time and less time to transmit the packet. A distance bounding scheme easily plugged into STAMP scheme. This is the future work to investigate such possibilities.

REFERENCES


