International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 13
October 2017

Enabling Integrity and Non-transferability of the Location proofs and protects users Privacy

Amburi Venkata Mounika & V Gopikrishna
[1]PG Scholar, Dept of CSE, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India.
[2]Assistant Professor, Dept of CSE, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India.

Abstract: *Services of Location based are quickly becoming hugely popular. The services based on users' current location, many potential services rely on users' location history, or their spatial-temporal Location. Malicious users are may lie about their spatial-temporal location without a carefully designed security system for users to prove their past locations. In this paper, we present the Spatial-Temporal provenance Assurance with Mutual Proofs (STAMP) scheme. STAMP is designed for MANET mobile users generating location proofs for each other in a distributed setting. It can provide easily to trusted mobile users and wireless access points. STAMP ensures the integrity and non-transferability of the location proofs and protects users' privacy. A semi-trusted Certification Authority is used to distribute cryptographic keys and guard users against collusion by a light-weight entropy-based trust evaluation approach. This prototype implementation on the Android platform shows that STAMP is low-cost in terms of computational and storage resources.*

Keywords: Enabling integrity, Non Transferability, Prototype

Most of the current location-based services for mobile devices are based on users' current location. Users discover their locations and share them with a server. In turn, the server performs
computation based on the location information and returns data/services to the users. In addition to users' current locations, there is an increased trend and incentive to prove/validate mobile users' past geographical locations.
Let us consider three examples: (1) A store wants to offer discountsto frequent customers. Customers must be able to showevidence of their repeated visits in the past to the store.
(2) Acompany which promotes green commuting and wellness mayreward their employees who walk or bike to work. The companymay encourage daily walking goals of some fixed number
of miles. Employees need to prove their past commuting paths to the company along with time history. This helps the company in reducing the healthcare insurance rates and move towards sustainable lifestyle.
(3) On the battlefield, when a scout group is sent out to execute a mission, the commanding center may want every soldier to keep a copy of their location traces for investigation Purpose after the mission. The above applications require users to be able to obtain proofs from the locations they visit. Users may then choose to present one or more of their proofs to a third-party verifier to claim their presence at a location at a particular time. In this paper, we define the past locations of a mobile user at a sequence of time points as the *spatial-temporal provenance*
(STP) of the user, and a digital proof of user's presence at a location at a particular time as an *STP proof*. Today's location-based services solely rely on users' devices to determine their location, e.g., using GPS. However, it allows malicious users to fake their STP information. Therefore, we need to involve third parties in the creation of STP proofs in order to achieve the integrity of the STP proofs. Location information is highly sensitive personal data. Knowing where a person was at a particular time, one can infer his/her personal activities, political views, health status, and launch unsolicited advertising, physical attacks or harassment [7]. Therefore, mechanisms to preserve users' privacy and anonymity are mandatory in an STP proof system.
Second, authenticity of STP proofs should be one of the main design goals in order to achieve integrity and non-transferability of STP proofs. Moreover, it is possible that multiple parties
Collude and create fake STP proofs. Therefore, careful thought must be given to the countermeasures against collusion attacks.

In this paper, we propose an STP proof scheme named Spatial-Temporal provenance Assurance with Mutual Proofs (STAMP). STAMP aims at ensuring the integrity and non-transferability of the STP proofs, with the capability of protecting users'

privacy. Most of the existing STP proof schemes rely on wireless infrastructure to create proofs for mobile users. However, it may not be feasible for all types of applications, e.g., STP proofs for the green commuting and battlefield examples certainly cannot be obtained from wireless APs. To target a wider range of applications, STAMP is based on a distributed architecture. Co-located mobile devices mutually generate and endorse STP proofs for each other, while at the same time it does not eliminate the possibility of utilizing wireless infrastructures as more trusted proof generation sources. In addition, in contrast to most of the existing schemes which require multiple trusted or semi-trusted third parties, STAMP requires only a single semi-trusted third party which can be embedded in a Certificate Authority (CA). We design our system with an objective of protecting users' anonymity and location privacy. No parties other than verifiers could see both a user's identity and STP information.Users are given the flexibility to choose he location granularity level that is revealed to the verifier.

We examine two types of collusion attacks: (1) A user who is at an intended location masquerades as another colluding user and obtains STP proofs for this attack has never been

addressed in any existing STP proof schemes.

(2) Colluding users mutually generate fake STP proofs for each other. There have been efforts to address this type of collusion. However,existing solutions suffer from high computational cost and low scalability. The experimental results show that STAMP requires low computational overhead.

The *contributions* of this paper can be summarized as:

1) A distributed STP proof generation and verification protocol (STAMP) is introduced to achieve integrity and non-transferability of STP proofs. No additional trusted third parties are required except for a semi-trusted CA.

2) STAMP is designed to maximize users' anonymity and location privacy. Users are given the control over the location granularity of their STP proofs.

3) STAMP is collusion-resistant. The Bussard-Bagga distance bounding protocol [9] is integrated into STAMP to prevent a user from collecting proofs on behalf of another user. An entropy-based trust model is proposed to detect users mutually generating fake proofs for each other.

4) STAMP uses a entropy-based trust model to guard users from prover-witness collusion. This model also encourageswitnesses against selfish behavior.

5) Modifications to STAMP to facilitate the utilization of stationary wireless infrastructure APs or trusted mobile users are presented.

6) A security analysis is presented to prove STAMP achieves the security and privacy objectives.

7) A prototype application is implemented on the Android platform. Experiments show that STAMP requires preferably low computational time and storage.

8) Simulation experiments validate that our entropy-based trust model is able to achieve over 0.9 collusion detection accuracy with fairly high percentage of colluding attackers.

The rest of the paper is organized as follows: Section II discussesrelated work. Section III describes our system model. In Section IV, we discuss the security requirements in detail and describe the threat model of this work. In Section V, we present the details of the STAMP protocol. Section VI provides an overview of how STAMP can be practically used in number of scenarios including trusted mobile users and wireless APs. In Section VII, we describe our implementation and simulation and present our experimental results on the performance evaluation. Finally, Section VIII concludes the paper.

II. RELATED WORK

The notion of unforgeable location proofs was discussed byWaters[10]. They proposed a secure scheme which advice can use to get a location proof from a location manager.However, it requires users to know the verifiers as a prior.Saroiu*et al.*

[1] proposed a secure location proof mechanism,where users and wireless APs exchange their signed public keys to create timestamped location proofs. These schemes are susceptible to collusion attacks where users and wireless APs may collude to create fake proofs.VeriPlace.

[2] is a location proof architecture which is designed with privacy protection and collusion resilience. However, it requires three different trusted entities to provide security and privacy protection: a TTPL (Trusted Third Party for managing Location in formation), a TTPU (Trusted Third Party for managing User information) and a CDA (Cheating Detection Authority). Each trusted entity knows either a user's identity or his/her location, but not both. VeriPlace's collusion detection works only if users request their location proofs very frequently so that the long distance between two location proofs that are chronologically close can be considered as anomalies. This is not a realistic assumption because users should have the control over the frequency of their requests. The system that is most closely related to our work is

Zhu *et al.*'s APPLAUS [3]. It is a location proof system that is also based on co-located mobile devices mutually generating location proofs. In order to protect privacy, the knowledge of private information is separately distributed to three parties: a location proof server, a CA, and the verifier. Periodically changed pseudonyms are used by the mobile devices to protect their real identities from each other, and from the location proof server. We believe the location proof server is redundant for accomplishing the goals. Periodically changed pseudonyms incurs high operational overhead because of the requirement for highly cautious management and scheduling. Dummy proofs have to be regularly generated in order to achieve the privacy properties, which also incurs high communication and storage overhead. The collusion detection in APPLAUS is based on betweenness ranking and correlation clustering. These approaches require the location proof server to have access to at least the majority of the concurrent (within a short delay) location proofs at the same location (within a small region). This needs users to submit their location proofs right after generating them, which is infeasible when there is no Internetconnection on-the-spot.

III. SYSTEM MODEL

Wireless infrastructure may not be available everywhere and hence a system based on wireless APs creating STP proofs would not be feasible for all scenarios. In addition, the deployment cost would be high if we require a large number of wireless APs to have the capability of generating STP proofs. Therefore, we think a distributed STP proof architecture, i.e., mobile users obtaining STP proofs from nearby mobile peers, would be more feasible and appropriate for a wider range of applications. We design a generic decentralized protocol, and then show how it can work well for centralized case also.Fig. 1 illustrates the architecture of our system.

There are four types of entities based on their roles:

• *Prover*: A *prover*is a mobile device which tries to obtain STP proofs at a certain location.

• *Witness*: A *witness* is a device which is in proximity with the prover and is willing to create an STP proof for the prover upon receiving his/her request. The witness can be untrusted or trusted, and the trusted witness can be mobile or stationary (wireless APs). Collocated mobile users are
untrusted.

• *Verifier*: A *verifier* is the party that the prover wants to show one or more STP proofs to and claim his/her presence at a location at a particular time.

• *Certificate Authority (CA)*: The *CA* is a semi-trusted server (untrusted for privacy protection, see Section IV-C for details) which issues, manages cryptographic credentials for the other parties. CA is also responsible for proof verification and trust evaluation.

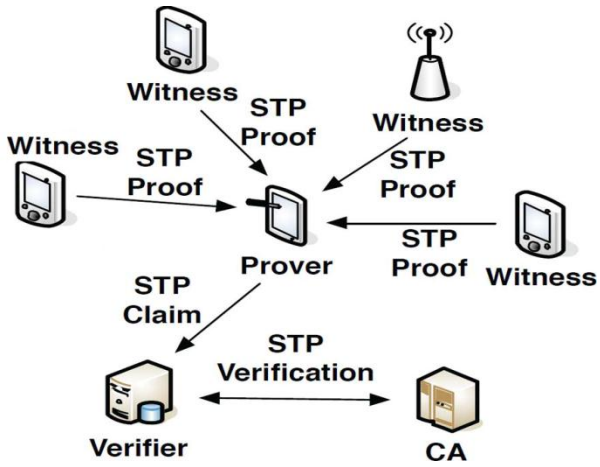

Fig 1 System Architecture

A prover and a witness communicates with each other via Bluetooth or WiFi in ad hoc mode. A peer discovery mechanism for discovering nearby witness is required and preferably provided
by underlying communication technology instead of our protocol. The proof generation system of prover is presented a list of available witnesses. When there are multiple witnesses willing to cooperate, the prover initiate protocol with them sequentially. STP claims are sent to verifiers from provers via a LAN or Internet, and verifiers are assumed to have Internet connection with CA. Each user can act as a prover or a witness, depending on their roles at the moment. We assume the identity of a user is bound with his/her public key, which is certified by CA. Users have unique public/private key pairs, which are established during the user registration with CA and stored onusers' personal devices. There are strong incentives for peoplenot to give their privacy away completely, even to their families or friends, so we assume a user never gives his/her mobile device or private key to another party.

IV. REQUIREMENTS AND CHALLENGES

Before introducing the details of our protocol, we first present and discuss the important issues and design challenges involved, in order to give an intuition of our objectives of constructing the protocol.

A. Security

The security of STP proofs are two fold: *integrity* and *non-transferability* . The integrity property requires that no prover can create fake STP proofs by himself/herself or by collaborating with one or more other untrusted parties in the system. The non-transferability property requires that no provercan claim the ownership of another prover's legitimate STP proofs.

B. Privacy

Anonymity: Location privacy is an extremely important factor that needs to be taken into consideration when designing any location based systems. Revealing both identity and location information to an untrusted party poses threats to a mobile users. First, a prover should be able to hide his/her identity from a witness. In addition, it is not only the prover's anonymity that we should pay attention to, a witness's anonymity should also be preserved. Since a witness who agrees to create an STP proof is co-located with the prover, his/her identity should not be revealed to the prover, either.

C. Threat Model

Prover: A malicious proverseeks to create fake STP proofs without physically being present at a location. This includes creating fake STP proofs by himself/herself, lying to a witness about his/her location, tampering with the spatial-temporal information in his/her existing proofs, as well as stealing and using another user's STP proofs. Moreover, a malicious prover also attempts to obtain a witness's identity information in the entire process of STP proof generation.

Witness: A malicious witness's goals include acquiring a prover's identity information and repudiating an STP proof that is generated by him/her.

Verifier: A verifier is often a service provider or an authority that is trying to validate a prover's STP claim. A prover has to present both his/her identity and STP information to the verifier in order to get a service or simply prove his/her alibi. We assume that a verifier is trusted in terms of privacy leakage, that is, a verifier never leaks a prover's identity or STP information to any other parties. However, a prover should be able to only give a verifier his/her STP information that is necessary. In other words, a prover should have the control over which STP proofs and what location granularity of the STP proofs are revealed to a verifier.

CA: We assume CA is trusted but curious, in the sense that it is only trusted in term of correctly performing its functions, i.e., user registration, key and credential management, and trust assessment for STP proofs. Also, CA does not intentionally leak any information it stores to other individual users. However, CA may intend to use any information it learned to profile user's spatial-temporal history and thus a potential privacy abuse may happen at CA.

Collusion: We specifically tackle two different collusion scenariosin this work:

(1) A witness can collude with a prover by creating an STP proof for him/her even though one or both of them are not at the location as claimed in the STP proof. We name this collusion scenario as *W-P collusion*. To the best of our knowledge, there is no good solution to detect this type of
collusion yet.

(2) A prover P who requires a colluding prover Qwho is at a specific location to masquerade as him/her and generate a fake STP proof. Though we assume does not give his/her private key to , it is possible for P and Q to have a hidden communication tunnel during the STP proof generation process, so that could relay messages to , signs on them and returns them to in real time. This kind of collusion attack is a type of *Wormhole* attack [13], which has been more commonly referred to as the *Terrorist Fraud* attack [8] in location verification. It is one of the most challenging attacks to protectagainst in location verification. Applied to our context, we name this collusion scenario as *P-P collusion*

V. THE STAMP SCHEME

A. Preliminaries

1) Location Granularity Levels: We assume there are ngranularity levels for each location, which can be denoted by ,Level1,Level2,Level3…… where $Level_1$represents the finest location granularity (e.g., an exact Geo coordinate), and $Level_n$ represents the most coarse location granularity (e.g., a city). Hereafter, we refer to location granularity level as *location level* for short. When a location level $Level_x$is known, we assume it is easy to obtain a corresponding higher location level$Level_y$ where y>x .The semantic representation of location levels are assumed to be standardized throughout the system.

International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 04 Issue 13
October 2017

2) Cryptographic Building Blocks: STAMP uses the concept of *commitments* to ensure the privacy of provers. A commitment scheme allows one to commit to a message while keeping it hidden to others, with the ability to reveal the committed value later. The original message cannot be changed after it is committed to. A commitment to a message M can be denoted as C(M,r)where r is a nonce used to randomize the commitment so that the receiver cannot reconstruct , and the commitment can later be verified when the sender reveals both M and r. A number of commitment schemes [14]–[16] have been proposed and commonly used. Our system does not require a specific commitment scheme. Any scheme which is perfect binding and computational hiding can be used. In our implementation, we used [14], which is based on one way hashing. One-way hash functions have the similar binding and hiding properties as commitment schemes. However, for privacy protection purpose, we do not use hash functions because they are vulnerable to *dictionary* attacks. An adversary who has a fulllist of possible inputs could run an exhaustive scanning over the list to crack the input of a hash function. We assume every user has the ability to generate one-time symmetric keys. All parties have agreed upon a one-way hash function and a commitment scheme. The commitment scheme is implemented based on any pseudo-random generator. All cryptographic notations have been summarized in Table I.

$M_1 \| M_2$	Concatenation of messages M_1 and M_2
K_u^+	Public key of user u
K_u^-	Private key of user u
$E^K(M)$	Encryption of message M with key K
$H(M)$	One-way hashing of message M
$C(M,r)$	Commitment to message M with nonce r

TABLE I

3) Distance Bounding: A location proof system needs a prover to be securely localized by the party who provides proofs. A distance bounding protocol serves the purpose. A distance bounding protocol is used for a party to securely verify that another party is within a certain distance [17]. Different types of distance bounding protocols have been studied and proposed. A most popular category is based on *fast-bit-exchange*: one party sends a challenge bit and another party replies with a response bit and vice versa. By measuring the round-trip time between the challenge and the response, an upper bound on the distance between the two parties can be calculated. This fast-bit-exchange phase is usually repeated a number of times.

B. Protocol

1) Overview: Our protocol consists of two primary phases: *STP proof generation* and *STP claim and verification.*

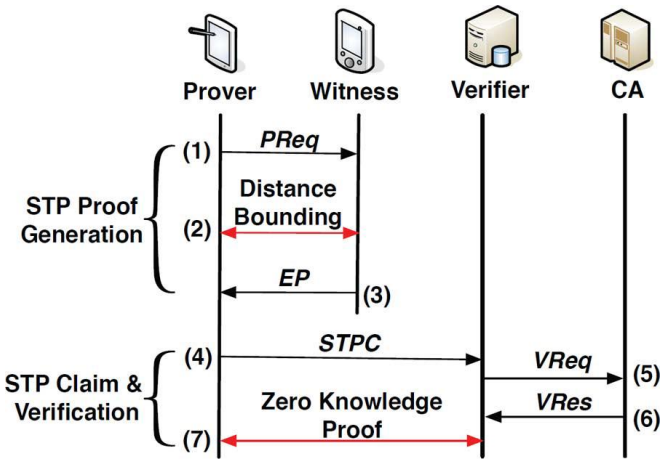

Fig. 2 gives an overview of the two phases and the majorcommunication steps involved.

When a prover collects STP proofs from his/her co-located mobile devices, we say an *STP proof collection event* is started by the prover. An STP proof generation phase is the process of the prover getting an STP proof from one witness. Therefore, an STP proof collection event may consist of multiple STP proof generations. The prover finally stores the STP proofs he/she collectedin the mobile device.When a prover encounters a verifier (the frequency of suchencounters is specific to the application scenarios) and he/she intends to make a claim about his/her past STP to the verifier, the STP claim and verification phase takes place between the proverand the verifier. A part of the verification job has to be done by CA. Therefore, communication between the verifier and CA happens in the middle of the STP claim and verification phase. In Fig. 2, the two arrowed lines in red color represent the latter two stages of the Bussard-Bagga protocol. These stages require multiple interactions between the two involved parties, and thereby are represented by doubly arrowed lines. The preparation stage of the Bussard-Bagga protocol does not need to be executed

for every STP proof generation and thus is not shown. Users could run the preparation stage before each STP proof collection event or pre-compute and store several sets of the bit commitments and primitives, and randomly choose one set of them when needed. Subsequently, we present the details of the STAMP protocol.

VI. USAGE & MODIFICATIONS

A. Selfish Node

Our proposed entropy-based trust model guards from P-W collusion by giving lower trust values to STP proofs generated by common or repeating witnesses. It also serves as an incentive mechanism for users to generate STP proofs for strangers. In a generic case, peer mobile users may be selfish. They may choose to save their battery power over generating STP proofs for other users, particularly when they are strangers.

B. Coarse Grain Location

Trust computation becomes more reliable with increased number of users, hence choosing a coarser location level may be preferable for those services which seek higher reliability and trust but lower location granularity. We now show how STAMP can be used to collate STP proofs from witnesses from different locations to verify coarse grain location with higher trust.

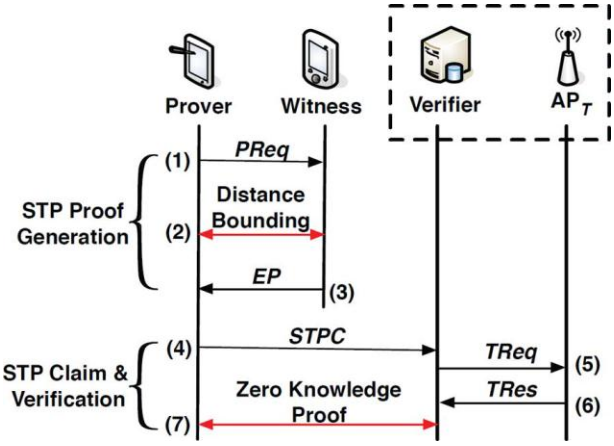

Fig 3: STAMP protocol with trusted wireless AP.

C. Trusted Witnesses

STAMP is useful for a wide range of application where a centralized infrastructure (trusted wireless APs) is not available. The green commuting application we described in Section I is a
good example scenario. In some scenarios, a trusted mobile or stationary user may be available or required. For example, a store which wants to give discounts to its frequent customers may have some trusted mobile users such as customer service agents who are amongst thecrowd in the store. In the prior case, we have incognito trusted mobile users. For users going to a park, it was observed that there are frequent events when users find no co-located user to generate STP proofs. Thus, the authorities set up a trusted wireless AP to generate STP proofs for travelers. The exact location of such trusted wireless AP is known. In these scenarios, the prover can send all to CA or skip using CA since the proofs are already trusted (Fig. 3). The first model fits well for incognito trusted mobile users while the other model serves well for wireless APs.

VII. EXPERIMENTS AND RESULTS

A. Prototype Implementation

We implemented a prototype client application on Android with Java. Our experiments are carried out on two Samsung Exhibit II 4G devices equipped with Qualcomm MSM 8255
1 GHz chipset, 512 MB RAM, 1 GB ROM, GPS, and Bluetooth, and running Android OS 2.3. Bluetooth is used as the communication interface between mobile devices. We use DSA key pairs for signing/authentication operations because DSA is based on the discrete-log problem, which makes it possess the mathematical properties desired by the Bussard-Bagga protocol.
Since DSA is not designed for encryption/decryption purpose, we use RSA key pairs as sub-keys for encryption/decryption operations. We use SHA1 as the one-way hashing function
and 128-bit AES as the symmetric key encryption scheme. We implemented the string commitment scheme presented in [14] and use it for ID and location commitments. We model each location with six levels: exact location, neighborhood, town/ city, region/county, state and country, where each level is represented by a name string except that the lowest level also has the
geo-coordinates.

1) Performance in Static Scenario: With our implementation, we examine the computational time (also an indicator of power consumption) and storage that are needed to run STAMP.
Since the STP verification is done by verifiers and CA where desktops or servers with high computational power can be used, we focus our testing on the STP proof generation phase that is
executed on mobile phones. The results we show are obtained based on 10 runs of each test. No other background processes were running in parallel during the tests.our client application. Since both DSA and RSA are used in our implementation, we test three key size combinations representing three different security levels:
(1) 512-bit DSA with 1024-bit RSA (denoted as 512/1024);
(2) 768-bit DSA with 2048-bit RSA (denoted as 768/2048);
(3) 1024-bit DSA with 3072-bit RSA (denoted as 1024/3072).
Figs. 4(a)–4(c) show the computational resources required with these three key size settings.

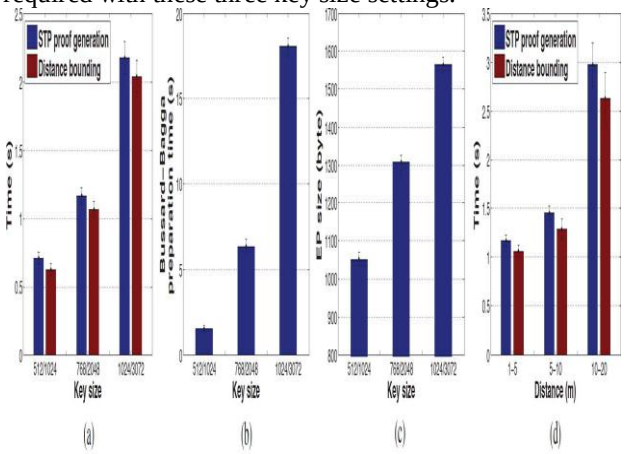

Fig. 4.Implementation results. (a) Time to generate an STP proof under different key sizes. (b) Time of Bussard-Bagga preparation stage under different key sizes. (c) Size of EP under different key sizes. (d) Time to generate an STP proof under different communication distances.

Fig. 4(a) shows the time needed for a prover to get an STP proof from a witness and for the portion of this process taken by the Bussard-Bagga distance bounding. We could observe that
a majority portion of the STP proof generation is taken by the Bussard-Bagga distance bounding. It is easy to disable the distance bounding stage for application scenarios where P-P collusion is not a concern. In that case, the time for each proof generation will be significantly reduced to less than 0.2 s even if large keys are used.
Fig. 4(b) shows the time needed for the Bussard-Bagga preparation stage. We can see this could cause long delay if large keys are used. However, as we explained in Section V-B, to achieve best unlinkability, it could be executed only once for an STP proof collection event. Under a relaxed unlinkability requirement, users could also pre-compute and store several sets of the bit commitments and primitives, and randomly choose one set of them when needed.
Fig. 4(c) shows the size of an that needs to be stored on a prover's mobile device. Since multiple could be received for each STP proof collection event, the size of an isthe main factor that determines the storage need for an STP proof entry. We can see that each is less than 2000 bytes. Though several such may need to be retained for each STP proof entry, the storage consumption is definitely acceptable considering the storage capacity of today's mobile devices.

Figs. 4(a)–4(c) tell us the choice of key size is critical. Larger keys provide stronger security, but also require more resources in terms of computational time and storage. For achieving general security requirements for a light-weight mobile application, we suggest using a small key size setting. In addition to key size, we study the impact of communication distance between mobile devices on the STP proof generation time.
Fig. 4(d) shows the results of our testing with the 768/ 2048 key size setting. As expected, the communication distance negatively affects the STP proof generation time as well as the distance bounding time.

2) Performance in Mobile Scenario: We are also interested in an alternative scenario when prover or/and witnesses are mobile. It evaluates the feasibility when our scheme is applied to location- based services with continuous tracking. We perform experiments in three typical mobility mode, namely *Walking (W)*, *Biking (B)*, and *Driving (D)*, with two speed levels respectively. An outdoor path of 45 meters long is used for all three modes, while an additional path of 161 meters is dedicated for driving test in high mobility. In each mode, the witness moves towards the stationary prover and then move pass him and away, while the scheme automatically scanning for available witnesses, establishingconnections, and generating location proofs. Since the protocol remains mostly the same as static scenarios, we focus more on qualitative metric of success rate instead of quantitative indicators which are similar to that of static scenario. Specifically,

experiments are repeated four times for each setting and ratio of successfully completion of protocol for at least once is reported. We choose a balanced key size setting of 768/2048 bits in all experiments.As expected, the performance isfairly stable in lower mobility involved in Walking and Biking settings. When walking at a low speed of 3.5 km/h, multiple location proofs can be successfully generated during one trip. We note that substantial time is consumed by Bluetooth inquiry and paging process for peer discovery, which takes approximately12 seconds, in addition to the actual proof generation process Nonetheless the results confirm Bluetooth connection provide adequate transmission range and is resilient enough in low and moderate mobility modes for our protocol to complete. On the other hand, when mobility level is increased the performance degraded drastically, as observed in driving tests. In our experiments all failures resulted from parties move out of transmission range before the inquiry process can complete. The intrinsic bottleneck of Bluetooth transmission range and overhead in discovery limits the performance of our scheme in high mobility scenarios.

VIII. CONCLUSION

In this paper we have presented STAMP, which aims at providing security and privacy assurance to mobile users' proofs for their past location visits. STAMP relies on mobile devices in vicinity to mutually generate location proofs or uses wireless APs to generate location proofs. Integrity and non-transferability of location proofs and location privacy of users are the main design goals of STAMP. We have specifically dealt with two collusion scenarios: P-P collusion and P-W collusion. To protect against P-P collusions, we integrated the Bussard-Bagga distance bounding
protocol into the design of STAMP. To detect P-W collusion, we proposed an entropy-based trust model to evaluate the trust level of claims of the past location visits. Our security analysis
shows that STAMP achieves the security and privacy objectives. Our implementation on Android smartphones indicates that low computational and storage resources are required to
execute STAMP. Extensive simulation results show that our trust model is able to attain a high balanced accuracy with appropriate choices of system parameters.

REFERENCES

[1] STAMP: Enabling Privacy-Preserving Location Proofs for Mobile Users Xinlei Wang, AmitPande, Jindan Zhu, and PrasantMohapatra, Fellow, IEEE-Dec 2014
[2] W. Luo and U. Hengartner, "VeriPlace: A privacy-aware location proof architecture," in Proc. ACM GIS, 2010, pp. 23–32.
[3] Z. Zhu and G. Cao, "Towards privacy-preserving and colluding-resistance in location proof updating system," IEEE Trans. Mobile Comput.,
vol. 12, no. 1, pp. 51–64, Jan. 2011.
[4] N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in Proc. ACM WiSe, 2003, pp. 1–10.
[5] R. Hasan and R. Burns, "Where have you been?secure location provenance for mobile devices," CoRR2011.
[6] B. Davis, H. Chen, and M. Franklin, "Privacy preserving alibi systems," in Proc. ACM ASIACCS, 2012, pp. 34–35.
[7] I. Krontiris, F. Freiling, and T. Dimitriou, "Location privacy in urban sensing networks: Research challenges and directions," IEEE WirelessCommun., vol. 17, no. 5, pp. 30–35, Oct. 2010.
[8] Y. Desmedt, "Major security problems with the 'unforgeable' (feige)-fiat-shamir proofs of identity and how to overcome them," in Proc.SecuriCom, 1988, pp. 15–17.
[9] L. Bussard and W. Bagga, "Distance-bounding proof of knowledge to avoid real-time attacks," in Security and Privacy in the Age of UbiquitousComputing. New York, NY, USA: Springer, 2005.
[10] B. Waters and E. Felten, "Secure, private proofs of location," Department of Computer Science, Princeton University, Princeton, NJ, USA, Tech. Rep., 2003.
[11] X. Wang et al., "STAMP: Ad hoc spatial-temporal provenance assurance for mobile users," in Proc. IEEE ICNP, 2013, pp. 1–10.
[12] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity-a proposal for terminology," in Designing PrivacyEnhancing Technologies. New York, NY, USA: Springer, 2001.
[13] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Wormhole attacks in wireless networks," IEEE J. Sel. Areas Commun., vol. 24, no. 2, pp. 370–380, Feb. 2006.
[14] S. Halevi and S. Micali, "Practical and provably-secure commitment schemes from collision-free hashing," in Proc. CRYPTO, 1996, pp. 201–215.
[15] I. Damgård, "Commitment schemes and zero-knowledge protocols," in Proc. Lectures Data Security, 1999, pp. 63–86.
[16] I. Haitner and O. Reingold, "Statistically-hiding commitment from any one-way function," in Proc. ACM Symp. Theory Comput., 2007, pp. 1–10.
[17] D. Singelee and B. Preneel, "Location verification using secure distance bounding protocols," in Proc. IEEE MASS, 2005.
[18] J. Reid, J. Nieto, T. Tang, and B. Senadji, "Detecting relay attacks with timing-based protocols," in Proc. ACM ASIACCS, 2007, pp. 204–213.

[19] C. Kim, G. Avoine, F. Koeune, F. Standaert, and O. Pereira, "The Swiss-knife RFID distance bounding protocol," in Proc. ICISC, 2009, pp. 98–115.
[20] H. Han et al., "Senspeed: Sensing driving conditions to estimate vehicle speed in urban environments," in Proc. IEEE INFOCOM, Apr. 2014, pp. 727–735.
[21] I. Afyouni, C. Ray, and C. Claramunt, "Spatial models for contextawareindoor navigation systems: A survey," J. Spatial Inf. Sci., no. 4, pp. 85–123, 2014.
[22] N. Roy, H. Wang, and R. R. Choudhury, "I am a smartphone and I can tell my user's walking direction," in Proc. ACM MobiSys, 2014, pp. 329–342.
[23] R. Steinbach, J. Green, and P. Edwards, "Look who's walking: Social and environmental correlates of children's walking in London," Health Place, vol. 18, no. 4, pp. 917–927, 2012.
[24] K. Brodersen, C. Ong, K. Stephan, and J. Buhmann, "The balanced accuracy and its posterior distribution," in Proc. IEEE ICPR, 2010, pp. 3121–3124.
[25] B. Peterson, R. Baldwin, and J. Kharoufeh, "Bluetooth inquiry time characterization and selection," IEEE Trans. Mobile Comput., vol. 5, no. 9, pp. 1173–1187, Sep. 2006.
[26] J. Zhu, K. Zeng, K.-H. Kim, and P. Mohapatra, "Improving crowdsourcedWi-Fi localization systems using Bluetooth beacons," in Proc. 9th Annu. IEEE SECON, Jun. 2012, pp. 290–298.

Authors Details:

AmburiVenkataMounika,
PG Scholar, Department of Computer Science and Engineering, PACE Institute of Tech and Sciences, Vallur, Ongole, AP, India..

V.GopiKrishna ,Assistant Professor, Department of Computer Science and Engineering, Completed his B.Tech from JNTU Kakinada and M.Tech is from Quba College of Engineering & Technology, Nellore,AP, India, His Experience is 7 Years. His Interested area is Network Security.