Australian Journal of Basic and Applied Sciences, 9(7) April 2015, Pages: 571-577

ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Propitious Privacy Preserving And Protection In Geo-Social Networks

R. Krithiga and V. Muthamilselvi

PG scholar, Department of Computer Science and Engineering, Valliammai Engineering College ,Kattankulathur, Kancheepuram District, Tamil Nadu, India.

ARTICLE INFO

Article history:
Received 12 December 2014
Received in revised form 26 December 2014
Accepted 28 January 2015
Available online 1 April 2015

Keywords:
Privacy preserving, location based service, Paillier cryptosystem, technology social factors, cloud computing.

ABSTRACT

Location-based service (LBS) is developing with the rapid growth of mobile devices and cloud computing model. It participates in Geo-social network additionally. Privacy-preserving LBS must be secure, provide accurate query and exposes user to a variety of privacy vulnerabilities. In this work we propose to take first steps toward addressing the conflict between profit and privacy in Geo-social networks. We tend to introduce a framework for constructing location centric profiles (LCPs) and composite built over the profiles of users that have visited separate locations. Additionally to a venue centric approach and we propose a decentralized solution for computing real time LCP snapshots over the profiles of collocated users. we propose a private circular query protocol to deal with privacy and accuracy issues of privacy preserving LBS in this protocol points of interest (POIs) and POI-related information (POI-info) are connected with zero knowledge about the user's location during the query process as a result the simulation shows that k-NN search query accuracy rate of the proposed protocol is very higher than 92% even when k is large.

To Cite This Article: R. Krithiga and V. Muthamilselvi., Propitious Privacy Preserving And Protection In Geo-Social Networks. ***Aust. J. Basic & Appl. Sci.,*** *9(7): 571-577, 2015*

INTRODUCTION

One of the popular services is LBS (e.g., Google Latitude), within which users will utilize the geographical information for gaining recreation services. On-line social networks became a big supply of non-public info. Their users voluntarily reveal a wealth of personal data, including age, gender, contact information, preferences and status updates. A recent addition to this space, geo social networks (GSNs) such as Yelp (yelp, 2014) and Foursquare (foursquare, 2014) further collects fine grained location information, through check-ins performed by users at visited venues. Yelp also trains small businesses to respond to reviews responsibly; hosts social events for reviewers; and provides basic data about businesses, such as hours of operation**.**

Foursquare aims to provide highly personalized recommendations of the best places to go around a user's current location (foursquare, 2014). venue owners to promote their businesses through spatial-temporal incentives and Providing personal information exposes however users to significant risks, as social networks have been shown to leak (B. Krishnamurthy and C. E. Wills,2010)and even sell user data to third parties.(E. Steel and G. Fowler, 2010).

In this work, we take first steps toward addressing this conflict. Our approach is based on the concept of location centric profiles (LCPs). LCPs are statistics built from the profiles of (i) users that have visited a certain location or (ii) a set of co-located use (iii) Paillier cryptosystem to deal with the privacy and the accuracy issues of privacy-preserving LBS.

Contributions We introduce a framework that allows the construction of LCPs based on the profiles of present users, while ensuring the privacy and correctness of participants. Informally, we define privacy as the inability of venues and the GSN provider to accurately learn user information, including even anonym zed location trace profiles. Verifying the correctness of user data is necessary to compensate for this privacy constraint: users may cheat and bias LCPs anonymously.

We consider two user correctness components. First Location correctness, where users should only contribute to LCPs of venues and where they are located. This requirement is imposed by the recent surge of fake check-ins (Foursquare Official Blog, 2011), motivated by their use of financial incentives. Second LCP correctness where users should be able to modify LCPs only in the predefined manner.

First, we propose a venue centric framework that relieves the GSN provider from a costly involvement

Corresponding Author: R.Krithiga, PG scholar, Department of computer science and engineering, Valliammai engineering college, Kattankulathur, Chennai, India.
E-mail: rkrithigarajesh@gmail.com.

in venue specific activities. To achieve this, stores and builds LCPs at venues. Furthermore, it relies on Paillier homomorphic cryptosystem. Second, we propose a completely decentralized extension, built around the notion of Metaphor LCPs. The distributed framework enables user devices to aggregate the profiles of co-located users, without assistance from a venue device. Snapshot LCPs are not bound to venues, but instead user devices can compute LCPs of neighbors at any location of interest. Communications and implementations are performed over ad hoc wireless connections. The contributions of this paper are then the following:

- Introduce the problem of computing location centric profiles (LCPs) while simultaneously ensuring the Privacy and correctness of participants.
- Propose a framework for computing LCPs. Devise both a venue centric and a decentralized Solution. Prove that satisfies the proposed privacy and correctness properties.

Provide two applications for: (i) privacy preserving, personalized public safety recommendations and (ii) privately building real time statistics over the profiles of venue patrons with Yelp accounts. Show that PROFIL*R* is efficient even when deployed on previous generation smart phones.

The remainder of the paper is organized as follows. First it describes the system and illustrative and defines the problem. Next it proves its privacy and correctness and finally it evaluates the performance of the proposed constructs, at last we concludes.

Illustrative And Background:

We consider a core functionality that is supported by the most influential geo social network (GSN) providers, Yelp (yelp, 2014) and Foursquare (foursquare, 2014). This functionality is simple and general enough to be applicable to most other GSNs (e.g., Facebook Places, Google Latitude). In this model, a provider *S* hosts the system, along with information about registered venues, and serving a number of users. To use the provider's services, a client application, the "client", needs to be downloaded and installed. Users register and receive initial service credentials, including a unique user id.

The provider supports a set of businesses or venues, with an associated geographic location (e.g., restaurants, yoga classes, towing companies, etc.). Users are encouraged to report their location, through *check-ins* at venues where they are present. During a check-in operation, performed upon an explicit user action, the user's device retrieves its GPS coordinates, reports them to the server, who then returns a list of nearby venues. The device displays the venues and the user needs to choose one as her current check-in location. Participating venue owners need to install inexpensive equipment (e.g., a $25 Raspberry PI (*Raspberry Pi*, 2012), a Beagle Board (G. Coley, 2009) any Android Smartphone and IOS Smartphone). This equipment can be installed and used for other purposes as well, including detecting fake user check-ins preventing fake badges (P. Paillier, 1999)and incorrect rewards, and validating social network (e.g., Yelp (yelp,2014)) reviews. Venue deployed equipment provides a necessary ingredient: ground truth information from remote locations.

A.LCP Requirements:

Let *k* is a security parameter, denoting the level of privacy we need to provide for users at any location. We then define a private LCP solution to be a set of functions,

PP (k) = {Setup, Spotter, Check In, Pub Stats},:

Setup is run by each venue where user statistics are collected, to generate parameters for user check-ins. To perform a check in, a user first runs *Spotter*, to prove her physical presence at the venue. *Spotter* returns error if the verification fails success otherwise. If *Spotter* is successful, *Check In* is run between the user and the venue, and allows the collection of profile information from the user. Specifically, if the user's profile value *v* on dimension *D* falls within the range *Ri*, the counter *ci* is incremented by 1. Finally, *Pub Stats* publishes collected LCPs. In the following, we use the notation *Port (P*1 *(args*1*)... Pn (argsn))* to denote protocol *Prot* run between participants *P*1... *Pn*, each with its own arguments. Let *CV* be the set of counters defined at a venue *V*. We use $\overline{C}V$ to denote the set of sets derived from *CV* as follows. Each set in $\overline{C}V$ differs from *CV* in exactly one counter, whose value increments the value of the corresponding counter in *CV*. For instance, if *CV* = {2, 5, 9}, then $\overline{C}V$ = {{3, 5, 9}, {2, 6, 9}, {2, 5, 10}}.
A private LCP solution needs to satisfy the following properties:
k- Privacy, Location Correctness, LCP Correctness, Check-In IN distinguish ability (CI-IND).

B. Location Centric Profiles:

A location centric profile is aggregates built over the profiles of users Present at a given location. Each user has a profile *PU*= {*pU*1, *pU*2... *pUd}*, consisting of values on *d* dimensions (e.g., age, gender, home city, etc.). Each dimension has a range, or a set of possible values. Given a set of users *U* at location *L*, the *location centric profile* at *L*, denoted by *LCP(L)* is the set {*LCP*1, *LCP*2, .., *LCPd* }, where *LCPi* denotes the aggregate statistics over the *I* –the dimension of profiles of users from *U*. In the following, we focus on a single profile dimension, *D*. We assume *D* takes values over a range *R* that can be discredited into a finite set of sub-intervals (e.g., set of continuous disjoint intervals or discrete values). Then, given an integer *b*, chosen to be dimension specific, we divide *R* into *b* intervals/sets, *R*1... *Rb*. For instance, gender maps naturally to discrete values (*b* = 2), while age can be divided into disjoint sub-intervals, with a higher *b* value. We define the aggregate statistics *S* for dimension *D* of *LCP(L)* to consist of *b* counters *c*1, .., *cb*; *ci* records the number of users from *U* whose profile value on dimension *D* falls within range *Ri* , *i* = 1..*b*.

C. Correlation attack and background knowledge attack:

The proposed secret circular shift is performed before each query and the amount of shift is determined only by the querying user, which can be regarded as a one-time pad encryption scheme, and therefore, providing high security. Servers cannot infer any knowledge about the user's location from the query history and the user's profiles, since the amount of shift has been scrambled by user and the POI information has also been encrypted. Under such circumstance, the Correlation Attack and Background Knowledge Attack made by the server cannot succeed.

We assume venue owners are malicious and will attempt to learn private information from their patrons. Clients installed by users can be malicious, attempting to bias LCPs constructed at target venues. We assume the GSN provider does not collude with venues, but will try to learn private user information.

D. Tools:

Homomorphic Cryptosystems:

The Paillier cryptosystem is the associated homomorphic property is sufficient to be integrated in the proposed protocol. Of course, further homomorphism properties, e.g., NTRU cryptosystem (D. Chaum, 1982), can assist in building a more efficient protocol.

Paillier Cryptosystem: Paillier cryptosystem (P. Paillier, 1999) is a public-key cryptography based on the decisional composite residuosity problem to guarantee its security. We will use E_r (m) to denote the encryption of a message $m \in z_n$ and $D(E_r$ (m)) to denote the corresponding decryption, a random number belongs to z_n and n , is a product of two large primes. The inherent random number r in the Paillier cryptosystem will prevent from generating the same cipher text of the same plaintext message, that is, the cipher text E_{r1} (m) will be totally different to the cipher text E_{r2} (m). In the field of secure computation, Paillier cryptosystem is famous for its Additive Homomorphism. That means, for a given public-keyk_p and the cipher texts $Er1(m)$ and E_{r2} (m). One can directly compute the addition of plaintexts in the encryption domain as:

$$D(Er_1(m_1).\ E_{r2}(m_2) \bmod n^2) = m_1 + m_2 \qquad (1)$$

Paillier also supports Homomorphic Multiplication of one cipher text and one plaintext, that is, for the given k_p, $Er1(m1)$ andm_2, one can directly compute the multiplication of m1*m2 in the encryption domain by

$$D((Er_1(m_1))^{m2}{}_{\bmod}\ n^{2)} = m1 * m2 \qquad (2)$$

With the aid of the two homomorphic properties, listed in (1) and (2), one can directly accomplish a matrix multiplication in the encryption domain as follows. Given a public-key k_p and two matrices, ($A_{i,\ k}$) (i=0,…, u-1; k==0,…, t-1) and ($B_{k\ ,j}$) (k=0,…, t-1; j=0,…,v-1). Now, encrypt every element of (A_i, k) into E_{rik} ($A_{i,\ k}$) but keep that ($B_{k\ ,j}$) intact. Without knowing the plaintext of (A_i, k), one can blindly do the encryption domain matrix multiplication of E_{rik} ($A_{i,k}$) and ($B_{k\ ,j}$) get the resultant encryption domain matrix multiplication ($C_{i,\ ,j}$) as

$$(C_{i,j}) = \qquad \prod_{k=1}^{t}(Er_{ik}(A_{i,k}))^{(B_{k,j})} \qquad (3)$$

The plaintext counterparts of ($c_{i,j}$) are still kept in secret.

Anonymizers: We use an anonymized (P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias,2007) that (i) operates correctly – the output corresponds to a permutation of the input and (ii) provides privacy – an observer is unable to determine which input element corresponds to a given output element in any way better than guessing. We use Orbot an Android implementation of Tor (orbot, 2013).

Proposed Method:

Let *Ci* denote the set of encrypted counters at *V*, following the *i* -the user run of *Check In*. *Ci* = {*Ci* [1], ..,*Ci* [*b*]}, where *Ci* [*j*] denotes the encrypted counter corresponding to *Rj*, the *j* -th sub-range of *R*. We write *Ci* [*j*] = *E(u j , u_j , c j , j)* = [*E(u j , c j),* *E(u_j , j)*], where *u j* and *u j* are random obfuscating factors and *E(u, M)* denotes the Paillier encryption of a message *M* using random factor *u*. That is, an encrypted counter is stored for each sub-range of domain *R* of dimension *D*.

The encrypted counter consists of two records, encoding the number of users whose values on dimension *D* fall within a particular sub-range of *R*.

Let *RE(v j , v_ j , E(u j , u_ j , c j , j)* denote the re-encryption of the *j* -th record with two random values *v j* and *v_ j* : *RE(v j , v_ j , E(u j , u_ j , c j , j))* = [*RE(v j , E(u j , c j)), RE(v_ j , E(u_ j , j))*] = [*E(u j v j , c j), E(u_ j v_ j , j)*]. Let *Ci* [j] ++ = *E (u j, u_ j, c j* +1*, j)* denote the encryption of the incremented *j* th counter. In the following we use the above definitions to introduce framework. Framework instantiates *PP (k)* and where *k* is the privacy parameter. The notation *P (A (paramsA), B (paramsB))* denotes the fact that protocol *P* involves participants *A* and *B*, each with its own parameters.

Setup (V (), S (k)): The provider *S* runs the key generation function *KG (l)* of the Paillier cryptosystem. Let *p* and *q* be the private key and *n* and *y* the public key. *S* sends the public key to SPOTR*V*. SPOTR*V* generates a signature key pair

and registers the public key with *S*. For each user profile dimension *D* of range *R* with *b* sub-ranges, SPOTR*V* performs the following steps:
Step 1: Initialize counters *c*1... *c b* to 0.
Step 2: Generate *C*0 = {*E(x*1, *x*_1, *c*1, 1*)*... *E (xb, x_ b, cb, b)*},

- Where xi, *x_i*, *i* = 1...*b* are randomly chosen values.
- Store *C*0 indexed on dimension *D*.
- Initialize the share set *Skey* = ∅.

Step 3: Generate system wide parameters *k* and *m* > *k* and initialize the *(k, m)* TSS.

Spotter (U (L, T), V(), S(k)): Let *L* and *T* denote *U*'s location and current time. To ensure anonymity, *U* generates fresh random MAC and IP addresses. These addresses are used for a single execution of the *Spotter* and *Check In* protocols. SPOTR*V* uses one of the location verification procedures proposed in (p.paillier, 1999) to verify *U*'s presence at *L* and *T* . Let *U* be the *i* -the user checking-in at *V*. If the verification succeeds and *i* ≤ *k*, *S* uses the *(k, m)* TSS to compute a share of *p* (Paillier secret key, factor of the modulus *n*). Let *pi* be the share of *p*. *S* sends the (signed) share *pi* to *U*. If *i* > *k*, *S* calls *Setup* to generate new parameters for *V*.

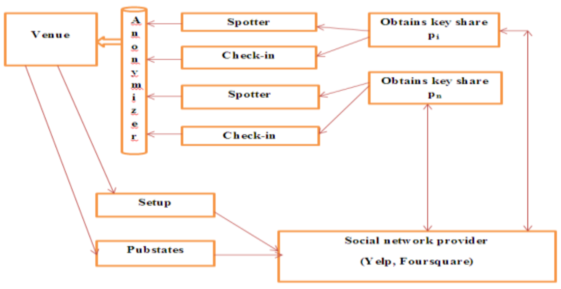

Fig. 1: System architecture (M= 3).

Check In (U (pi, n, V), V (n, y, Ci−1, Skey)): Executes only if the previous run of *Spotter* is successful. *U* uses the same random MAC and IP addresses as in the previous *Spotter* run. Let *U* be the *I* th user checking-in at *V*. Then, *Ci*−1 is the current set of encrypted counters. SPOTR*V* sends *Ci*−1 to *U*.

Let *v*, *U*'s value on dimension *D*, be within *R*'s *j* -th sub range, i.e., *v* ∈ *RJ*.
U runs the following steps:
• Generate *b* pairs of random values {*(v*1, *v*_1*)*... *(Vb, v_ b)*}.

- Compute the new encrypted counter set *Ci*,
- Where the order of the counters in *Ci* is identical to *Ci*−1:
- *Ci* = {*RE(vl, v_ l ,Ci*−1[*l*]*)*|*l* = 1..*b*, *l* _= *j*} ∪ *RE(v j , v_ j ,Ci*−1[*j*] ++*)*}.

Send *Ci* and the signed (by *S*) share *pi* of *p* to *V*.

If SPOTR*V* successfully verifies the signature of *S* on the share *pi*, *U* and SPOTR*V* engage in a zero knowledge protocol.

ZK-CTR: ZK-CTR allows *U* to prove that *Ci* is a correct re-encryption of *Ci*−1: only one counter of *Ci*−1 has been incremented. If the proof verifies, SPOTR*V* replaces *Ci*−1 with *Ci* and adds the share *pi* to the set *Skey*. Otherwise, SPOTR*V* drops *Ci* and rolls back to *Ci*−1.

Pub Stats (V (*Ck*, Sh, V), S(p ,q)): SPOTR*V* performs the following actions:
• If |*Sh*| < *k*, abort.
• If |*Sh*| = *k*, use the *k* shares to reconstruct *p*, the private Paillier key
• Use *p* and *q* = *n/p* to decrypt each record in *Ck*, the final set of counters at *V*. Publish results.

ZK-CTR: Proof of Correctness:

We now present the zero knowledge proof of the set *Ci* being a correct re-encryption of the set *Ci*−1, i.e., a single counter has been incremented. Let ZK-CTR (i) denote the protocol run for sets *Ci*−1 and *Ci*. *U* and SPOTR*V* run the following steps *s* times:
• *U* generates random values *(t*1, *t*_1*)*... *(tb, t_ b)* and random permutation *π*, then sends to SPOTR*V*
the proof set *Pi*−1 = *π* {*RE (tl, t_ l, Ci*−1 [*l*]*), l* = 1...*b*}.
• *U* generates random values *(w*1, *w*_1*)*... *(Wb, w_ b)*. It sends to SPOTR*V* the proof set *Pi* = *π* {*RE (wl, w_ l, Ci* [*l*]*)*, *l* = 1...*b*}
• SPOTR*V* generates a random bit *a* sends it to *U*. If *a* = 0, *U* reveals random values *(t*1, *t*_1*)*, *(tb, t_ b)*
and *(w*1, *w*_1*)*... *(Wb, w_b)*. SPOTR*V* verifies that for each *l* = 1...*b*, *RE(tl , t_l ,Ci*−1[*l*]*)* occurs in *Pi*−1
exactly once, and that for each *l* = 1...*b*, *RE (wl, w_ l ,Ci* [*l*]*)* occurs in *Pi* exactly once.
• If *a* = 1, *U* reveals *ol* = *vlwl t*−1 *l* and *o_l* = *v_lw_l t_*−1 *l*, for all *l* = 1...*b* along with *j*, the position in *Pi*−1 and
Pi of the incremented counter. SPOTR*V* verifies that for all *l* = 1...*b*, *l* _= *j*, *RE (ol, o_ l, Pi*−1 [*l*]*)* = *Pi* [*l*] and
RE (o j , o_ j , Pi−1[*j*]*y)* = *Pi* [*j*].
• If any verification fails, SPOTR*V* aborts the protocol.

Preventing Venue-User Collusion (Sybil attack):

The Sybil attack has appeared in many forms in both academic work and in the real world. It is a severe and pervasive problem in many areas. For example, it is possible to rig Internet polls by using multiple IP addresses to submit votes, to gain advantage in any results of a chain letter (A. Khoshgozaran, C. Shahabi, and H. Shirani- Mehr, 2011) and is a well-known and potentially major problem in real-world elections. A Sybil attack is also used by companies that increase the Google Page Rank rating of the pages of their customers (P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, 2007) and has been used to link particular search terms to unexpected results for political commentary. Reputation systems are a common target for Sybil attacks including real-world systems like eBay.

The use of the anonymized prevents the provider and the use of the unique IP and MAC addresses prevents the venue from differentiating between interactions with the same or different accounts. In this section we propose a solution, that when used in conjunction with Sybil detection tools, mitigates this problem. The solution introduces a trade-off between privacy and security. Specifically, we divide time into epochs (e.g., one day long). A user can check-in at any venue at most once per epoch. When active, once per epoch *e*, each user *U* contacts the provider *S* over an authenticated channel. *U* and *S* run a blind signature protocol: *U* obtains the signature of *S* on a random value, *RU, e*. *S* does not sign more than one value for *U* for any epoch. In runs of *Spotter* and *Check In* during epoch *e*, *U* uses *RU, e* as its pseudonym (i.e., MAC and IP address). Venues can verify the validity of the pseudonym using *S*'s signature. A venue accepts a single *Check In* per epoch from any pseudonym, thus limiting the user's impact on the LCP. The privacy breach mentioned above is due to the fact that now *S* can correlate *Check In* executed using the same *RU, e*. However, *S* does not know the real user identity behind *RU, e* – due to the use of blind signatures. (X. Pan, X. Meng, and J. Xu, 2009)

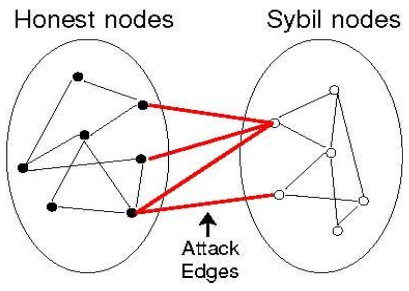

Fig. 2: Sybil attack.

Experimental Evaluation:

For testing purposes we have used a Dell laptop equipped with a 2.4GHz Intel Core i5processor and 4GB of RAM for the server and Samsung Admire smart phones running Android OS Gingerbread 2.3 with an 800MHzCPU .For local connectivity the devices used their 802.11b/g Wi-Fi interfaces. All reported values are averages taken over at least 10 independent protocol runs. We have first measured the overhead of the *Setup* operation .If *d* is the number of profile dimensions, *N* is the Paillier modulus size and *b* the sub-range count of domain *D*, the computation overhead of Setup is *TSetup* = *Tkeysig*+*dbTE* +*TT SS*.*Tkeysig* is the time to generate the signature key, *TE* is the average time of Paillier encryption and *TT SS* is the time to initialize the TSS (i.e., random polynomial generation).The storage overhead of *Setup* is *Store Setup* = d*bN*.

The computation overhead of *Check In* is *TCI* = *bTRE* + *TZK*, where *TRE* is the Paillier re-encryption cost and *TZK* is the overhead of the ZK-CTR protocol. The formula does not consider the cost of modular multiplication, random number generation and random permutation operations that are negligible compared to the other costs. Given *s*, the number of rounds of ZK-CTR, *TZK* = 2*sbTRE* + *sbTRE* + *s*2 *bTRE* = 72 *sbTRE*. The communication overhead is *Tcom_ CI* = *bN* + *T com_ ZK*.
The communication cost of ZK-CTR, *T com _ZK* is
S (2*bN* + 1
2 4*bN* + 12
2*bN)* = 5*sbN*.

We now focus on the most resource consuming component, the ZK-CTR protocol. While the above formulas assume similar capabilities for the client and venue components, we now measure the client side running on the Smartphone and the venue component executing on the laptop. Fig. 4 shows the dependence of the three costs for a single round of ZK-CTR on the Paillier modulus size. Given the more efficient venue component and the superior computation capabilities of the laptop, the venue component has a much smaller overhead. We have set *b* = 10. The communication overhead is the smallest, exhibiting a linear increase with bit size. For a Paillier key size of 1024 bits, the average end-to-end overhead of a single ZK-CTR round is 148ms. The venue component is29ms and the client component is 106ms. Furthermore, the overheads of these components as a function of the number of ZK-

CTR rounds, when the Paillier key size is 1024 bit and b = 10. For 30 rounds, when a cheating client's probability of success is 2−30 (1 in a billion), the total overhead is 3.6s (B. Carbunar, M. Rahman, J. Ballesteros, and N. Rishe, 2012).We further examine the communication overhead in terms of bits transferred during ZK-CTR between a client and a venue.

The communication overhead in a single ZK-CTR round is$4bN$ + $3bN$ = $7bN$. The second component of the sum is due to the average outcome of the challenge bit. Fig. 6 shows the dependency of the communication overhead (in KB) on b, when N = 2048. Even when b = 10, the communication overhead is around 21KB. The storage overhead is only a fraction of the (single round) communication overhead $7BN$. For a single dimension, with 30 sub-ranges, the overhead is 7KB. (S. A. V. Alfred, J. Menezes, and P. C. van Oorschot, 1996).

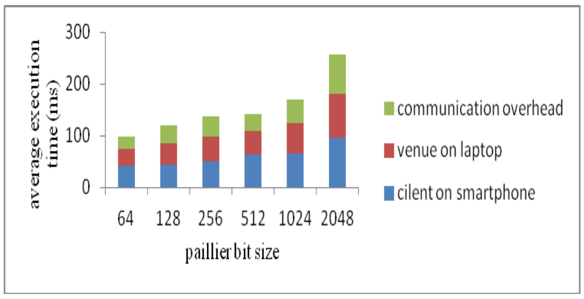

Fig. 3: The overhead imposed by ZK-CTR as a function of the Paillier modulus size.

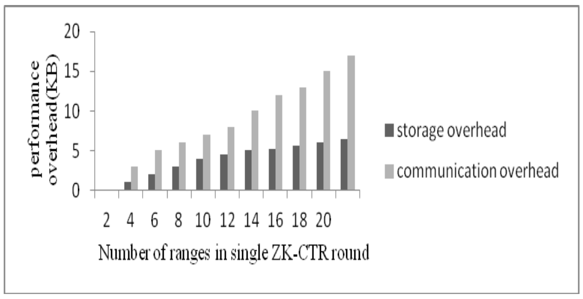

Fig. 4: Storage and Communication overhead in statistics computation.

As results this framework is applied in two different applications, i.e. public safety and real time yelp venue states. Public safety is a function of context of the space and presences. This greatly influenced by the people present in that particular space, this framework is divided in to a set of users to privately and correctly computes the distribution of their safety index values.

Conclusion:

In this paper, we have proposed a framework and mechanisms for privacy and correctly building location centric profiles. We have proved the ability of our solutions to satisfy the privacy and correctness requirements using different applications. We have shown that framework is efficient, even when executed on resource constrained mobile devices. The scalability of the framework is very powerful, when comparing with other homomorphic protocols .Furthermost this framework can also be used in cloud computing models. The proposed framework not only can address the challenges of privacy preserving LBS but also inspire the research of secret computations.

.

REFERENCES

Alfred, S.A.V., J. Menezes, P.C. van Oorschot, *1996. Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press.

Carbunar, B., M. Rahman, J. Ballesteros and N. Rishe, 2012. "Private location centric profiles for Geosocial networks," in *Proc. 20th ACM SIGSPATIAL Int. Conf. Adv. Geo graph. Inf. Syst.*, 458-461.

Chaum, D., 1982, "Blind signatures for untraceable payments," in *Proc. Adv. Cryptol. CRYPTO*, 199-203.

Coley, G., 2009. *Beagle board System Reference Manual*. Dallas, TX, USA, BeagleBoard.Org.

Foursquare Official Blog, 2011. New York, NY, USA. *On Foursquare, Cheating, and Laiming Mayor ships from your Couch* [Online]. Available: http://goo.gl/F1Yn5.

Foursquare, 2014. New York, NY, USA. [Online].Available: https://foursquare.com.

Kakhki, A.M., C. Kliman-Silver and A. Mislove, 2013. "Iolaus: Securing online content rating

Systems," in *Proc. 22nd Int. World Wide Web Conf.(WWW)*, Rio de Janeiro, Brazil, 1-5.

Kalnis, P., G. Ghinita, K. Mouratidis and D. Papadias, 2007. "Preventing location-based identity Inference in anonymous spatial queries," *IEEE Trans. Knowl. Data Eng.*, 19(12): 1719-1733.

Khoshgozaran, A., C. Shahabi and H. Shirani-Mehr, 2011. *Location Privacy: Going Beyond k-Anonymity, Cloaking and Anonymizers*. New York, NY, USA: Springer-Verlag New York, Inc., 26(3): 435-465.

Krishnamurthy, B. and C.E. Wills, 2010. "On the leakage of personally identifiable information via online social networks," *Compute. Commune. Rev.*, 40(1): 112-117.

Olumofin, F.G., P.K. Tysowski, I. Goldberg and U. Hen Gartner, 2010. "Achieving efficient Query privacy for location based services," in *Proc .Privacy Enhancing Technol.*, 93-110.

*Orbot,2013. ToronAndroi*d [Online].Available:https://www.torproject.org/docs/android.html.end.

Pan, X., X. Meng and J. Xu, 2009, "Distortion-based anonymity for continuous queries in Location-based mobile services", in *Proc.*, 256-265.

Paillier, P., 1999. "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology Euro crypt 1999*. New York, NY, USA: Springer-Verlag, 223-238.

Raspberry Pi, 2012. An ARM GNU/Linux Box for $25. Take a Byte [Online].Available: http://www.raspberrypi.org.

Steel, E. and G. Fowler, 2010. *Facebook in Privacy Breach* [Online].Available: http://online.wsj.com/article/SB100014240527023047728045755584840752369%68.html.

Yelp, 2014. Inc., San Francisco, CA, USA.[Online]. Available: http://www.yelp.com.