Verifiable and Privacy-preserving Check-ins for Geo-social Networks

Yonglei Yao[1,2], Jiezhong Zhu[1,2], Jingfa Liu[1,2], Neal N. Xiong[3]

[1] School of Computer and Software, Nanjing University of Information Science & Technology, China
[2] Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science & Technology, China
[3] School of Computer Science, Colorado Technical University, USA
ylyao@nuist.edu.cn, Jzzhu@nuist.edu.cn, jfliu@nuist.edu.cn, nxiong@coloradotech.edu

Abstract

Enabling efficient and accurate location verification is a desirable technique for the proliferation of many "check-in" based location sharing services. Existing solutions either use modified infrastructures, such as WiFi APs and cellular towers, to issue location proofs for users, which suffer from high infrastructure extension cost, or rely on co-located users to issue location proofs for each other, which are vulnerable to collusion attacks. In this paper, we propose a novel location verification approach that is based on the concept of location tag. When a user requests to check-in at certain venue, both the device of the user and of the venue will submit a location tag, which actually is a collection of environmental signals presented at a certain location and time, to the service provider. Based on the similarity of the two location tags, the service provider can verify whether the user is actually at the location as she claimed. Due to the high time- and space- dependence of environmental signals, it is hard for users to forge location tags. On the other hand, to protect privacy of a user from the location-sharing service, periodically changed pseudonyms are used by users when checking-in at venues, and partially blind signatures are employed by the services to issue check-in confirmation messages. Then users will submit the check-in confirmation messages in a batch manner, using their true Identities. Thus, the location-sharing service knows who has checked-in at which venue, but does not know when the user did. As a result, temporal privacy of user check-ins can be achieved. Thorough analysis and experiments are conducted to demonstrate the feasibility, security and efficiency of the proposed approach.

Keywords: Check-ins, Privacy, Location verification

1 Introduction

In recent years, mobile devices such as Smartphones and PDAs, have been playing an increasingly important role in our lives. With the proliferation of these location-enabled mobile devices, location-based services (LBSs) have become very popular. In LBSs, mobile devices discover and report their locations to a service provider, which in turn takes advantage of the location information to provide mobile users with certain type of resource or service. Typical location-based applications, to list a few, include points of interest (POIs) finders such as Qype [1], which help a mobile user to find a POI, friend finder such as Loopt [2], which search all friends in the vicinity of a user, and geo-social networks(GSNs) such as Facebook Places [3] and Foursquare [4].

Among various location-based applications, geo-social networks are a recent but increasingly popular one, and have developed as a "killer application". GSNs center their functionality on the relations between users and registered businesses, also called venues. Users "check-in" at a venue, such as a bar, a restaurant, etc., share their current position with friends, or leave recommendations. To promote their businesses, as a common theme, venues or the service provider would like to provide rewards to users who check in more frequently at these venues. For example, stores would offer discount coupons, restaurants could provide priority seatings, and so on. Geo-social networks have attracted more and more users. Sites like Foursquare and Facebook Places have garnered tens of millions of registered user accounts.

Similar to other LBSs, there are various security and privacy concerns associated with GSNs, which may prevent their widespread adoption. A major concern is how to prevent service abuse by malicious users who report forged locations so that to check-in at a venue far away from where they really are. As virtual or real-world rewards are provided to users checking-in at a certain venue, malicious users have the incentive to do so. The work in [5] showed the feasibility of fake check-ins in Foursquare, where a user explores the open source operating systems to modify the GPS-related APIs and cheat using falsified GPS information. The root cause of this vulnerability is the lack of

*Corresponding Author: Yonglei Yao; E-mail: ylyao@nuist.edu.cn
DOI: 10.3966/160792642018081904001

proper location verification mechanisms, as most mobile devices have the capability of discovering their locations, but they lack a mechanism to prove the locations to the services. Thus, it is essential to provide GSN providers the ability to verify user locations when they check-in, so as to ensure the social welfare of GSNs.

Several approaches have been proposed to provide the GSN providers the ability to verify user locations. Under the assumption that support from the wireless network infrastructure is available, approaches such as [6-7] propose to let certain trusted entities with fixed geolocations, namely WiFi APs or cellular towers, issue location proofs to users. However, these approaches require extensions of WiFi APs and cellular towers, and as a result will incur heavy deployment overhead. Another line of work, such as [8, 9], follows a peer-to-peer approach, where co-located Bluetooth enabled mobile devices issue location proofs for each other. However, these approaches require multiple users be present at the same location and same time. In addition, multiple malicious users may collude to issue falsified location proofs.

Another major concern related to GSNs is user privacy. In current GSNs, by check-ins performed by each user, the service provider can know who has visited which venue, and the exact time that she visited. With the information, the service provider can learn user activity pattern, such as location visiting order, dependencies between different locations, and so on. Thus, the service provider may profile users in a fine-grained manner and infer sensitive information. Most previous work on location verification has omitted this privacy issue.

In this paper, we propose a location verification approach for check-ins in GSNs, also with temporal privacy protection of user check-ins. For location verification, when a user checks-in at a venue, the service provider will request both user device and venue device to submit a location tag, i.e., a collection of environmental signals. Based on the similarity of features extracted from these signals, the service provider can determine whether the user is actually in the venue she claimed. The rationale behind the approach is that, signals received by two devices that in close proximity are highly correlated. We also observe that, for a GSN to fulfill their functionality, it needs to know the identity and spatial information of check-ins, i.e., who has visited which venues. However, the temporal information of the check-ins, which specifies the point of time that a user visited a venue, is not necessary. Thus, we propose an anonymous check-in approach, in which users use pseudonyms when checking-in at a venue, and update their check-in data using the true identifies later. As a result, the service provider cannot know when a user visited the venues, i.e., the temporal privacy of user check-ins is protected.

The main contributions of this paper are as follows.

(1) We propose a novel user location verification approach for GSNs in which environmental signals collected by user device and venue device are compared to determine whether the user is indeed in the venue she claimed. We demonstrate our concept using collected real-world WiFi signals, to show the feasibility of the proposed approach.

(2) We enhance the location verification approach with certain level of user privacy protection. Periodically changed pseudonyms are used by users when checking-in at a venue, and true identities are used when updating check-in data in a batch manner. As a result, the temporal privacy of user check-ins is protected.

(3) We carry out thorough performance evaluation using a proof-of-concept implementation. The results show the efficacy and efficiency of our approach.

The rest of the paper is organized as follows. Section 2 presents the problem formulation. Section 3 introduces the environmental signal-based user location verification approach and Section 4 describes anonymity-based user privacy protection approach. In Section 5, we conduct experiments to evaluate the performance of our approaches. Section 6 summarizes related work and Section 7 concludes.

2 Problem Formulation

2.1 System Model

We consider a high-level architecture for geo-social networks (GSNs) illustrated in Figure 1. The architecture consists of three types of entities: a service provider, a large number of users, and a set of venues.

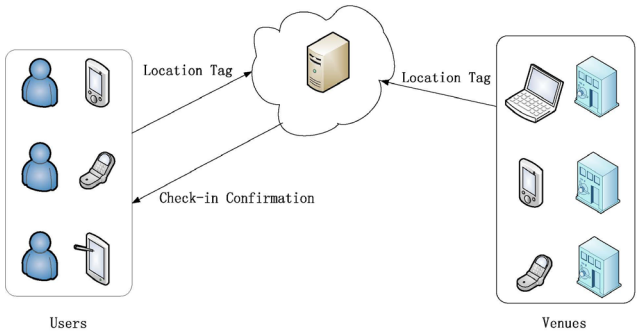


Figure 1. System architecture

Users are subscribers of the geo-social networking service operated by the service provider, and venues are various business entities, such as shopping malls, cafe stores, restaurants, etc. Both users and venues own some devices that can communicate with the service provider and collect environment signals, for example RF radios including WiFi, cellular, TV. The typical devices include Smartphones, PDAs, tablets, and laptops. Each venue will register her device when joining the GSN.

When visiting the venues, users may perform check-in, share the check-in data with their friends, or leave recommendations and comments. To encourage user check-in and boost their businesses, venues or the service provider will provide virtual or real-world rewards to users who perform frequent check-ins at a venue. For convenience, we use U to refer to a user, and V to denote a venue.

To prevent malicious users from cheating, when a user U requests to check-in at a venue V, the service provider will verify whether the user is indeed in the venue that she claimed. To enable the location verification, both U and V will construct a location tag, which is actually a collection of environmental signals, and sent the location tag to the service provider. The service provider measures the similarity of the two location tags, and if deriving a positive result, replies user U with a check-in confirmation message.

Another issue is to protect the temporal privacy of user check-ins, without interference with the GSN functionality, to prevent the service provider from profiling users in a fine-grained manner. By temporal privacy, we mean that the service provider can know who have visited which venues, but cannot know the exact time that these visitings happened. To achieve this, when checking-in at venues, users will use periodically updated pseudonyms, and the service provider employ partially blind signature in constructing confirmation messages. Later in some time, a user will update her check-in data in a batch manner, i.e., submit all the check-in confirmation messages that she has received in certain past period to the service provider.

2.2 Threat Models

In this paper, we consider several most serious attacks for GSNs.

Location cheating attack. Some users may be malicious. A user may check-in with a falsified location, to get the rewards without actually visiting the venues. This will result in money loss for venues.

Self-promotion attack. Also, malicious users may try to corrupt the system to self-promote themselves. For example, with pseudonyms used in the check-in process, when visiting a venue, a user may perform multiple check-ins with different pseudonyms. In addition, self-promotion may be achieved by replaying old check-in confirmation messages to increase the count of check-ins at certain venue without contributing new visiting. Again, venues will lose money because of this type of malicious behavior.

User profiling attack. The GSN service provider is considered as "*honest-but-curious*". Specifically, the service provider will follow the designed protocol specification to let users check in at different venues. However, it is curious to infer and analyze the detailed information of user check-ins, to profile users in a fine-grained manner. The "*honest-but-curious*" model is consistent with real world business practice. As the service provider can make profit by providing services, it is of interest for it to behave honestly to earn a high reputation and attract more users. On the other hand, the service provider also has a great interest to track and profile users, as user profile information can be used in accurate advertising to make profit.

2.3 Design Goals

Our designed approach should simultaneously achieve several security and performance goals as follows.

Location authentication. Our approach should prevent malicious users from checking-in at a venue using a falsified location.

Temporal privacy of check-ins. Our approach should provide temporal privacy of user check-ins, by not leaking the precise visiting time of user check-ins which is not necessary for the service provider to perform its functionality, thus preventing fine-grained user profiling by the service provider.

Self-promotion resistance. Our approach should prevent malicious users from self-promoting. Specifically, a user can check-in using only one pseudonym at certain time, and a check-in confirmation message should only be accepted by the service provider only once.

Efficiency. Above goals on security and privacy should be achieved with low communication and computation overhead.

3 Location Verification

In this section, we present the location verification approach based on location tag [12], which in essence is a collection of environmental signals. Our location verification approach compares the similarity of location tags submitted by user device and venue device, to determine whether both devices are in close physical proximity. Our approach leverages two key observations: first, two devices in close proximity will perceive similar environmental signals, while devices that are separated by a large distance will not. Thus, based on the similarity of collected environmental signals, we can determine whether user device and venue device are in close proximity. Second, due to the complexity of an environment, it is very difficult, if not impossible, to predict the nature of the environmental signals at certain time, which means the cost to forge a location tag is very high.

3.1 Overview

Our location verification protocol is a two-step protocol with which the service provider (SP) can determine whether the user device is actually in close proximity of the venue device, as shown in Figure 2.

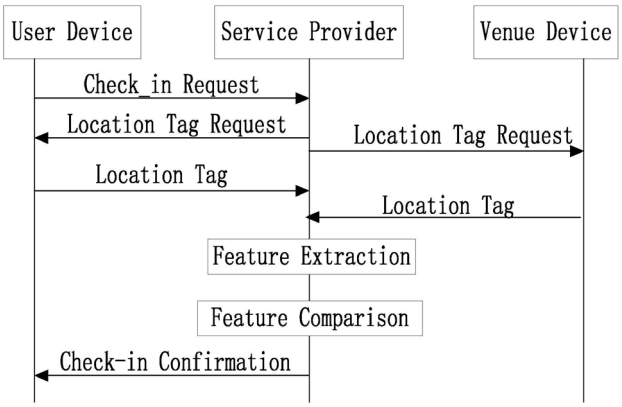


Figure 2. Flowchart of the location verification process

During the first step, the user device sends a *check-in request* message to the service provider, which includes her pseudonym (introduced in the next section) and the venue identity. After authenticating the user, the service provider will request both the user device and the venue device to collect a location tag. During the second step, upon receiving the location tags, the service provider extracts a set of features from the two tags respectively, and then compares the two extracted feature sets. If the similarity of the two feature sets goes beyond a threshold, the service provider determines the user is actually at the venue, and replies a *check-in confirmation* message to the user.

Note that, as user device and venue device are resource-restricted ones, such as Smartphones and tablets, while the service provider owns resource-rich servers and usually are supported by Clouds, we push as much work as possible to the service provider in the location verification process. Specifically, the user device and the venue device only collect and send raw environmental signals, without any processing on the collected signals. The service provider will extract features from the submitted raw signal, which often is a computation-intensive work and incurs heavy overhead.

3.2 Location Tag

There are many potential sources of location tags, such as WiFi, cellular, TV, FM broadcast, background noise, to list a few. In this paper, we use WiFi signals to form location tags, because of its near-ubiquitous nature. WiFi interfaces have become a standard component of most devices, including Smartphones, PDAs, Tablets, etc. In addition, WiFi interfaces can measure the signal strength information of the received frames as part of its normal operation. Mainly employing received signal strength (RSS) as features of location tag, our location verification protocol thus is a pure software solution without any interference with the wireless network infrastructure.

When receiving the *location tag request* message, a device D, either of the user U or venue V, will perform multiple passive wireless scans for all available WiFi access points (APs). Here, instead of a single scan, multiple scans are used to tolerate noisy radio signals and beacon losses.

For each available AP, the device records the SSID of the AP, the sequence number of each frame, and the corresponding RSSI. Specifically, for each AP, the device records the following signal trace:

$$< SSID^D, \{< SN_j^D, rssi_j^D >\} >, 1 \le j \le n \qquad \textbf{(1)}$$

Where $SSID^D$ denotes the SSID of an AP detected by D, SN_j^D denotes the sequence number of the j-th frame scanned by D, and $rssi_j^D$ is the RSSI of the j -th frame.

User device and venue device then submit the location tag, i.e., all the collected signal traces, to the service provider. Note that, due to the different positions of user device and venue device, the AP sets detected by each is usually not the same, i.e., $\{SSID^U\}$ is not the same as $\{SSID^V\}$. In addition, as wireless channel generally is a multi-path channel influenced by many factors such as reflection and scattering, a slight perturbation in the environment can generate significant signal fluctuations at the receivers. Thus, even for an AP detectable for both user device and venue device, the frame sequence detected by the both devices are not the same, e.g., some frames may temporarily undetectable for a device.

3.3 Determining Proximity

With the location tags submitted by the user device and venue device, the service provider determines whether they are in proximity in three steps. First, based on the number of APs common to both devices, the service provider determines to either reject the check-in request or proceed with the verification process. Second, based on the two signal traces, one collected by the user device and the other by venue device, from a same AP, a decision is made. Third, all these decisions based on traces from all the APs are combined to make a final decision.

Pre-matching decision. In the first step, the service provider determines whether the user device and venue device are not in close proximity by a simple indicator. Specifically, the service provider computes the following *Common AP Ratio*:

$$CAR = \frac{\left|\{SSID^U\} \cap \{SSID^V\}\right|}{\left|\{SSID^U\} \cup \{SSID^V\}\right|} \qquad \textbf{(2)}$$

If this ratio is below a threshold, δ_{AP}, the service provider determines that the user device is far away from the venue device, and rejects the user's check-in request. The rationale behind the decision is simple: if both devices are in close proximity, they will detect a set of common APS, as WiFi APs usually have a range

of tens of meters indoors and even hundred meters outdoors.

Single decision. In the second step, for each AP in the common AP set $\{SSID^U\} \bigcap \{SSID^V\}$, a decision is made by the service provider.

First, the service provider computes

$$SN = \{SN^U\} \bigcup \{SN^V\}$$
$$m = |SN| \quad (3)$$

Where SN^U and SN^V are the set of sequence numbers of frames collected from the same common AP by the user device and venue device respectively. Note that, m is definitely greater than the length of each signal trace.

Second, the service provider builds two RSS vectors with length m for user and venue:

$$RV^U = \{rssi_k^U\}$$
$$RV^V = \{rssi_k^V\} \quad (4)$$
$$1 \le k \le m$$

The rules of generate each element of the two vectors are as follows.

Assume the signal trace of the user device for the AP is $\{< SN_j^U, rssi_j^U >\}, 1 \le j \le n$. If there exist j and k such that $SN_j^U = SN[k]$, which means that the user signal trace contains the information on the frame with sequence number *SN[k]*, then we get $RV^U[k] = rssi_j^U$. Applying this rule on all the RSS values in the signal trace, we set the n elements in RV^U. However, as described above, even the two devices are in the same venue, one device may omit some frames detected by the other. Thus, there are *m-n* elements are not defined yet. We apply the interpolation technique to set these elements

Finally, the service provider uses the Pearson correlation to compute the similarity of the two RSS vectors. The Pearson Coefficient has the property of measuring the variations and not the absolute values themselves. Thus, this metric works well in our context, where two heterogeneous devices exist. The range of Pearson similarity values are from -1 to 1: a value of -1 means that the two data sets vary inversely, while 1 indicates that the sets are perfectly correlated. With a high Pearson value, we would be able to conclude that the user device and venue device are in proximity.

The Pearson similarity r is computed as follows:

$$\frac{\sum_{i=1}^{m}(RV^U[i] - \overline{RV^U})(RV^V[i] - \overline{RV^V})}{\sqrt{\sum_{i=1}^{m}(RV^U[i] - \overline{RV^U})^2}\sqrt{\sum_{i=1}^{m}(RV^V[i] - \overline{RV^V})^2}} \quad (5)$$

Where $\overline{RV^U}$ and $\overline{RV^V}$ denotes the mean value of the RSS values in RV^U and RV^V, respectively.

A threshold δ_R is set for r. If r is greater than this threshold, the service provider makes a decision that the two devices are in close physical proximity, based on the signal traces collected from a single AP.

Combined decision. In the final step, the service provider will combine all the decisions made based on signal traces collected from all the common APs. In the simplest voting scheme, if at least 50% of these single decisions give positive answers, then the service provider determines that the user device is indeed in close proximity with the venue device, and the user is actually in the venue as she claimed.

4 Anonymous User Check-ins

In this section, we present the approach to hide the temporal information of user check-ins. We first provide an overview of the cryptographic primitives used in construction of the approach, and then present our anonymous check-in mechanism.

4.1 Cryptographic Primitives

Blind signature. A blind signature scheme [10] enables a user to obtain a signature on a message from a signer but without revealing the message to the signer at the time of signing. Specifically, to obtain a signature on her message m without revealing it, the user first blinds m with a random blinding factor and gets a blinded message m'. The blinded message m' is then sent to the signer. With a standard signature algorithm, the signer signs on m' and returns the signature ϕ' to the user. The user then removes the blinding factor from ϕ' and obtains the signature ϕ on m.

Blind signature schemes satisfy two security-related properties. The first property is *blindness*, i.e., the signer cannot link $<m, \phi>$ to m' or ϕ'. The second one is *unforgeability*, which ensures that the user can not forge a signature from ϕ' for a message other than m.

In this paper, we do not assume any specific blind signature scheme. We use r to denote the random blinding factor, and $m*r$ the blind operation performed by the user. We also use *B_SIG(m)* to denote the blind signature on message m.

Partially blind signature. Similar to bind signature, a partially blind signature scheme [11] also allows a user to obtain a signature on message m without revealing it to the signer. The difference is that, the signer can explicitly include some common information, which is under agreement with the user, in the signature. If the common information is included in many signatures, the issuer cannot link a signature to a specific message. In addition, the *unforgeability* property also holds for partially blind signature.

We also do not assume any specific partially blind signature scheme. We use *PB_SIG(p,m)* to denote the

partially blind signature on message *m*, where *p* denotes the common information.

4.2 Anonymous Check-in

We assume that each user *U* has a permanent identity, denoted as *TID*, which is the id that the user registered with the service provider. In addition, each user *U* is assumed to have a long-term private/public key pair (d_U, e_U). We also assume that the time is divided into multiple equal-length periods. A typical time period *T* is a day, or alternatively a week.

The proposed scheme includes three steps, which are repeated sequentially in each time period *T*.

Pseudonym generation. For each time period *T*, the service provider generates a new private/public key pair (d_T, e_T), and broadcast the key pair to all the registered users.

At the end of previous time period or the start of the current one, the user *U* will generate a new pseudonym P_T, which is only valid in period *T*.

The user *U* first generates a random number, which will be used as the pseudonym P_T. The pseudonym needs to be confirmed by the service provider using blind signature. To do so, the client prepares another random number *r* as the blinding factor used in blind signature, and blinds the pseudonym P_T as follows:

$$m_P = P_T * r \tag{6}$$

Note that, * denotes the blind operation, as presented above.

Then *U* signs on the concatenation of m_P, *TID* and *T:*

$$Sig_{m_P} = SIG_{d_U}(m_P \| TID \| T) \tag{7}$$

The user *U* sends m_P, Sig_{m_P}, *TID* and the time interval *T* to the service provider *SP*, which together serve as a pseudonym application message:

$$U \rightarrow SP : m_P, Sig_{m_P}, TID, T \tag{8}$$

The service provider *SP* first verifies whether the user *U* has applied a pseudonym for the current time period *T*; if the answer is yes, *SP* will reject the application request. Otherwise, *SP* verifies the authenticity of m_P based on Sig_{m_P} and the public key of *U*. After the verification, *SP* signs m_P using a blind signature algorithm and the signature key of time period *T*, and sends the signed message back to *U*:

$$SP \rightarrow U : B\_SIG_{d_T}(m_P) \tag{9}$$

Once receiving the message, *U* removes the blinding factor from $B\_SIG_{d_T}(m_P)$ and obtains the signature on the pseudonym P_T, $SIG_{d_T}(P_T)$.

User checking-in. During time period *T*, the user *U* will use the pseudonym P_T as her identity when checking-in at a venue *V*.

U first generates a random number, which will serve as the identity of the current check-in, *CID*. Then *U* generates another random *r* and blinds *CID* with *r*. Finally, *U* sends the following check-in request message to the *SP*:

$$U \rightarrow SP : P_T, SIG_{d_T}(P_T), V, CID * r \tag{10}$$

After receiving the check-in request message, the service provider *SP* first verifies the validity of the pseudonym P_T using key e_T. If successful, *SP* knows that P_T is a valid pseudonym for certain user, because *SP* has signed on it. Then the service provider requests both the user device and the venue device to submit location tags, and performs the location verification process described in the previous section. After a successful location verification, *SP* signs on $CID * r$ using a partially blind signature algorithm, with *V* as the common information in the partially blind signature. *SP* sends the signature back to *U*, which serves as a check-in confirmation message:

$$SP \rightarrow U : PB\_SIG_{d_T}(V, CID * r) \tag{11}$$

When receiving the confirmation message, *U* removes the blinding factor and obtains the signature on *V* and *CID*. Then *U* stores $< T, V, CID, PB\_SIG_{d_T}(V, CID) >$, which serves as a check-in proof.

Check-in update. When obtaining a check-in proof, the user *U* cannot submit it to the *SP* immediately to update her check-in data at a venue, as this will help *SP* to link the pseudonym with the true identity easily. Instead, *U* will store the check-in proofs locally.

After some random length of time, *U* can submit all the check-in proofs in a batch manner to *SP*.

The check-in update message is as follows:

$$\begin{aligned} &U \rightarrow SP : \\ &TID, \{< T, V, CID, PB\_SIG_{d_T}(V, CID) >\} \end{aligned} \tag{12}$$

Where the first part is the true identity, and the second part is a collection of check-in proofs.

The service provider will verify the validity of each check-in proof. By *T*, *SP* knows e_T will be used as the verification key. By *CID*, *SP* can determines whether this check-in proof is a fresh or replayed one. After the verification, *SP* will update the check-in data of user *U* with true identity *TID*.

4.3 Security Analysis

Temporal privacy of check-ins. We use entropy as the metric of temporal privacy of user check-ins. Without pseudonym, a user checks-in at a venue with her true identity. As the service provider can know the exact time of the check-in, the temporal privacy is *0*.

When checking at a venue, a user *U* submits the check-in request message using her pseudonym, which

is blindly signed by the service provider. Due to the *blindness* property of the blind signature scheme, the service provider cannot link the pseudonym with a true identity. As a result, at the time of checking-in, the service provider can know one of the registered users is doing this, but cannot know her true identity.

When updating the check-in data, a user U submits her true identity and a collection of check-in proofs. However, a check-in proof is derived based on the partially blind signature on a check-in identity. Again because of the *blindness* property, the service provider cannot link a check-in identity with a specific check-in request. Thus, the service provider cannot determine when this check-in happens.

So our scheme ensures that the service provider can obtain the information required for its functionality, i.e., who visits which venue, but cannot know the exact time that the visiting happened. Suppose the service provider has some additional statistical information on each venue, i.e., the probability that a user visits the venue V at certain time t, P_v^t, satisfying $\sum P_v^t = 1$. Thus, given a check-in proof at venue V, the temporal privacy is

$$H = -\sum_t P_v^t \log P_v^t \quad \textbf{(13)}$$

With our periodical pseudonym, the temporal privacy of each check-in is increased from *0* to H. The service provider cannot determine the precise visiting time of each check-in, the order of all the check-ins, or the dependence between the check-ins. Thus, fine-grained profiling of a user becomes very difficult, if not impossible.

Self-promotion resistance. In our scheme, the time is divided into time periods, and in each period, a user must apply a new pseudonym. In the pseudonym application process, a user must show her true identity. Thus, in each time period, each user can apply only one pseudonym. In addition, the service provider uses a different key for each period to sign pseudonyms. If a user uses an old pseudonym in current period, it cannot pass the verification, because of the unforgeability property of digital signature. As a result, each user can only own at most one pseudonym when checking-in at venues, and sybil attack is prevented.

Our scheme also ensures that a check-in proof can be accepted by the service provider only once. This because, each check-in proof contains a *ID*, which is generated by the user and partially blindly signed by the service provider. Without the signed id included in the location proof, the service provider will not accept the check-in update request. By checking the freshness of the check-in *ID*, check-in proof replay can be prevented, and a user cannot use a check-in proof multiple times when updating her check-in data.

In our scheme, each check-in proof also contains the venue ID, which is included as the common information when blindly signed by the service provider. As a result, a user can not update her check-in data of a venue using check-in proofs obtained when visiting other venues.

5 Evaluation

To evaluate the feasibility and performance of our proposed approaches, we have built a proof-of-concept implementation. The implementation consists of three components: a user device, a venue device, and a service provider. The user device is a Samsung Galaxy S4 I9508 phone, which is equipped with a 1.9GHz quad-core CPU, 2GB RAM, a 2600mAh battery, and runs Android 4.2 OS. A Nexus 7 tablet acts as the venue device, which has a quad-core CPU with 1.3GHz frequency, 1GB RAM, a 4325mAh battery, and runs Android 4.1 OS. The service provider is implemented on a Windows laptop with a i5-3210M Dual-Core CPU, 4GB RAM, and running Win7 SP1 x64 OS.

5.1 Performance of Location Verification

Our experiments are conducted in a 2-storey building in our campus. The building has a dimension of about 65m by 40m, and includes 15 rooms at the ground floor and 30 rooms the second floor.

We conduct the experiments in two different settings. In the first setting, we place the venue device, i.e., the Nexus 7 tablet, at a fix location at the ground floor. We choose other several locations around the same floor, each with different distance with the tablet. Then someone carries the user device, i.e., the Sumsang Galaxy s4 Smartphone, visiting all the selected locations one by one. When arriving at a selected location, the Smartphone sends a check-in request to the laptop acting as a service provider. Then the laptop requests both the tablet and the Smartphone to collect locations. Both devices will perform passive WiFi scans for 5 seconds, and transmit the collected signal traces to the laptop.

In the second setting, the venue device is placed at the same place as in the first experiment. However, we choose the user device locations at the second floor, to test the impact of walls and floors.

Impact of distance on pre-matching decision. After receiving the two location tags, the service provider will make a preliminary decision solely on the common APs ratio, as introduced in Section 3.3.

In our experiments, the venue device can detect 34 APs in total. Table 1 shows the information of APs detectable by the user device when placed at the same ground floor as the venue device, with different distances.

Table 1. Information of detectable APs at the same floor

Distance(m)	5	10	15	20	30	35	40
Number of Common APs	24	23	22	20	20	22	22
Number of Union APs	44	45	46	48	48	49	48

Roughly speaking, when the distance increases, the common APs detectable by both device will decrease (Location No.7 with distance *40m* is an exception), consistent with our expectation.

When the user device is placed at locations in a different floor with the venue device, the information of APs detectable is shown in Table 2. Due to the effect of obstacles (walls and floors), the number of common APs detectable by the venue device with the same distance but at a different floor is slightly less than that at the same floor.

Table 2. Information of detectable APs at different floors

Distance(m)	10	15	20	30	35	40
Number of Common APs	22	22	21	19	18	16
Number of Union APs	47	48	48	50	49	47

With these data in hand, we calculate the common APs ratios (CARs) with different distance. The result is shown in Figure 3.

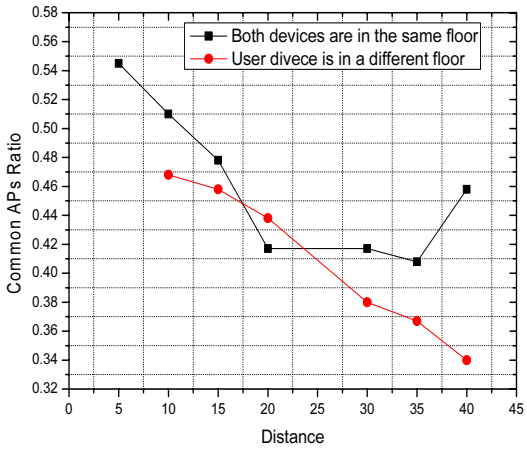


Figure 3. Impact of distance on common APs ratio

In the building, setting δ_{AP} to be 0.3 is safe, i.e., with false negative rate *0*. If the common APs ratio is below this threshold, it has a high probability that the distance between the devices is larger than 40m, which indicates that the user is unlikely in the building.

Impact of distance on single decision. For each AP in the common AP set, the service provider will calculate the Pearson similarity of the two RSS vectors collected by the venue and user devices. Taken a certain AP as an example, the impact of distance on the Pearson similarity is shown in Figure 4.

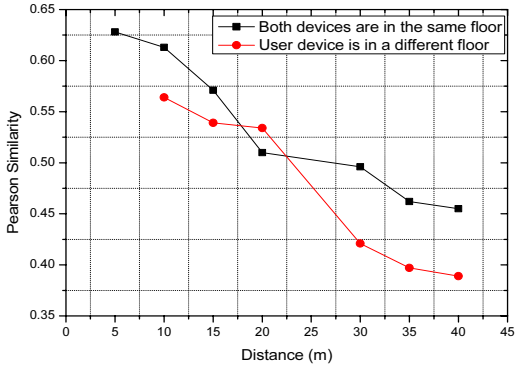


Figure 4. Impact of distance on pearson similarity

As the figure shows, roughly speaking, when the distance between the venue device and the user device increases, the Pearson similarity between the RSS vectors collected by them will decrease. This is consistent with the observation that, close devices will perceive similar signal fluctuations while distant devices will not.

In the building that we conduct experiments, for decision based on this AP, a threshold 0.355 is appropriate. Of course, the threshold δ_R varies on different APs. In our experiments, we find that, for each common AP, with a threshold about 0.22, the false negative rate of the single decision is *0*.

Verification effectiveness. As a final step, a majority voting is used to make a final decision. To measure the effectiveness of our verification approach, we use two metrics: false positive rate and false negative rate. False positive rate denotes the probability that a cheating user can pass the verification, which measures the soundness of our approach. In contrast, false negative rate is the probability that an honest user is denied, which measures the correctness of the approach.

In our experiments, to make a final decision based on majority voting, a value about 0.38 for δ_R is appropriate, as the false positive and false negative rate are the lowest. With this threshold, the false negative rate and false positive rate with different distances are shown in Table 3.

Table 3. Verification accuracy

Distance	≤20	30	35	40	>40
False Positive Rate	-	-	-	8.7%	4.6%
False Negative Rate	-	-	4.2%	9.5%	-

Thus, at the boundary of the building, the false positive rate and false negative rate are the highest. While in or distant from the building, both rates are satisfactory.

Remark. Different venues will have different RF radio features. When joining a geo-social networks, a venue should engage in an enrollment process, to let the service provider learn appropriate parameters for the

venue.

5.2 Efficiency of Anonymous Check-ins

In this subsection, we analyze and evaluate the efficiency of our anonymous check-in scheme. As service providers are usually resource-rich, we only pay attention to cost of mobile user devices, with respect to the metrics of energy consumption.

Our proof-of-concept implementation uses 1024-bit RSA and partially blind RSA [30] as the blind signature and partially blind signature scheme, respectively. Under the RSA blind digital scheme, assume that the private/public key pair is *(d, n)* and *(e, n)*, the blind operation is as follows:

$$m_B = m.r^e \mod n \quad \textbf{(14)}$$

Where m denotes the message to be blindly signed, r is the blinding factor, and m_B denotes the blinded message.

Then the signer blindly signs the message as follows:

$$B\_SIG(m) = m_B \mod n \quad \textbf{(15)}$$

The client removes the blinding factor and obtains the signature on m:

$$SIG(m) = B\_SIG(m).r^{-1} \mod n \quad \textbf{(16)}$$

The partially blind RSA signature operates in a similar way.

We consider a simple situation where a user uses a pseudonym for only one check-in. The computations performed by user device include the generation of a blind RSA signature and a regular signature in the pseudonym generation step, and a partially blind RSA signature in the check-in step. Thus, the total computation cost for a user device mainly includes 5 modular exponentiations, 5 modular multiplications and 5 modular inversions. The computation cost of the service provider is two signatures, which include 5 modular exponentiations.

As to the communication, a user device sends a *pseudonym application message*, a *check-in request message*, and a *check-in update message*, and receives a blindly signed pseudonym, a *check-in confirmation message*. Assuming that the user identity *TID*, the time period identifier *T*, the pseudonym, the venue identifier *V*, and the check-in identifier *CID* each has a 128-bit length, and supposing 1024-bit RSA is used, the communication cost of the user device is 4352-bit sending and 2048-bit receiving in total.

We measure the cost of the user device in terms of energy consumption. In the evaluation, the Smartphone iteratively runs the three steps in anonymous check-in, i.e., pseudonym generation, check-in request, check-in update (note that, without the location tag collection step), for 1000 times. We measure the energy consumed for the 1000 runs, and use the average consumption as an indicator of energy consumption of each run. Table 4 shows the cost, i,e., energy consumption of the privacy protection for user device.

Table 4. Verification accuracy

	Pseudonym Generation	Check-in Request	Check-in Update
Computation	430 *mj*	320 *mj*	
Communication	67 *mj*	64 *mj*	41 *mj*
SubTotal	497 *mj*	384 *mj*	41 *mj*
Total		922 *mj*	

Our anonymity-based temporal privacy protection approach consumes about *922mJ* for each check-in. For a Smartphone with a 3.7V, 2600mAh battery that contains about *34600J* energy, the consumption is negligible.

6 Related Work

6.1 Location Verification

One of the first to investigate location cheating attacks is the work in [5], which uses Foursquare as an example to show the feasibility of location cheating. Recently, Polakis et al. [13] create a platform for testing the feasibility of fake-location attacks in Foursquare and Facebook Places. Instead of tampering with GPS readings as in [5], they shows how the application layer of LBS can be leveraged to transmit fake information. These results show that current cheating-detection mechanisms adopted by popular geo-social networks are not adequate.

Many approaches have been proposed to allow service providers to validate user locations. Generally speaking, prior works can be grouped into two categories, depending on the presence or absence of the infrastructure.

In the infrastructure-dependent approaches, there exists a centrally-operated entity for each venue which issues location proofs for users. For instance, a WiFi AP executes a distance bounding protocol [14-15] to measure the distance between the user and the AP, and issues a location proof if the distance is below a threshold. Following the similar idea, a user proves her proximity by including the sequence number of a AP beacon [6] or a nonce broadcasted by a AP or a cellular tower [7] in her location proof request, and applies a location proof from the AP or the cellular tower. The recent work in [16] develops a beacon-based infrastructure that uses Bluetooth Low Energy signals to provide room-level location. However, all these approaches need the support from the infrastructure, which will incur heavy deployment cost. The work in [13] also provides a proof-of-concept location validation scheme based on NFC, in which a NFC device is deployed in each venue. As admitted by the authors, a cost of over $50 will be incurred for each

venue. In addition, the temporal privacy of users can not be protected.

In the infrastructure-independent approaches, users obtain location proofs from neighbors through short-range communication, such as Bluetooth [8-9, 17]. Talasila et al. [9, 17] propose LINK, in which a user obtains location certification messages from neighbors, and sends the messages to a Location Certification Authority (LCA). Based on the trust level of each neighbor, the LCA determines the authenticity of the user's location claim. Similarly, Zhu et al. [8] propose APPLAUS, in which co-located mobile devices mutually act as witness and generate location proofs relying on Bluetooth communications. Then these location proofs are sent to a centralized location proof server. Later, a verifier and an online Certificate Authority (CA) collaboratively retrieve location proofs and verify their authentication. However, these approaches are vulnerable to collusion attacks, in which malicious users generate fake location proofs for each other. To address this problem, Khan et al. [18] develop a hybrid approach and introduce a collusion-resistant protocol. In their approach, a user obtains location proof from a local location authority, which is also asserted by other witness. However, this approach requires a location authority deployed at each venue, and cannot protect the temporal privacy of user visiting, as timestamp is an important factor used in fighting against collusion attack.

In contrast to prior works, our location verification approach exploits the near-ubiquitous RF radios and mobile devices to generate location tags, instead of relying on the functionality-extended WiFi APs and cell towers, or dedicated local location authorities. In addition, our approach does not rely on the co-located witness, and as a result provides high usability and collusion-resistance.

6.2 Location Privacy

Privacy is an important concern in recently emerged computing paradigms, including cloud computing (e.g. [32-33]), mobile computing (e.g. [31]) and Social Networks (e.g. [34]), as sensitive information is stored and processed by these platforms.

Various approaches have been proposed to protect user location privacy, an important issue in the context of mobile computing, to protect user from tracking and profiling. Most of these approaches employ location perturbation or obfuscation techniques. Many approaches are based on the *k*-anonymity model, such as [19-21], to list a few. The basic idea is to hide a user's location among other *k-1* locations. In approaches based on the mix-zone model, such as [23], a user is assigned a new pseudonym each time entering a new mix zone, to hide the user trajectory. In the location cloaking technique such as [24, 28], instead of a accurate location point, a cloaking region will be reported to the server. Chen et al. propose a selecting and shuffle technique [29], in which only a subset of points from a real user trace will be selected, shuffled and reported to the server, to protect user trace privacy. With these approaches, the server cannot determine the accurate location of the user, but a perturbed one. Another line of work addresses the access control of location data, such as [22, 25-27]. In these works, as a user's location data is encrypted or transformed, only the user herself or authorized friends can query the location data, while the server is completely oblivious to these data. However, in the context of "check-in" services, service providers need the fine-grained location information of users to perform their functionality. Thus, these approaches cannot be applied to our context, as they significantly ruin the utility.

In addition to location verification, Luo et al. [7] also address the user anonymity issue. When requesting location proof from a issuer, a user will use a pseudonym generated with the help of the issuer. However, the goal is only to prevent multiple issuers from tracking users, and the service provider can still obtain the detailed temporal and spatial information of user activities. Similarly, APPLAUS [8] also uses periodical pseudonyms. The adversaries considered include the CA, the witness, and the location proof server, but not the location verifier. Thus, anyone acting as a location verifier can obtain the detailed information of user visiting. In contrast, our privacy protection scheme focuses on the temporal privacy of user visitings with respect to all the other entities, including the service provider. In the context of activity summaries, the work in [16] achieves both location authentication and privacy. As stated above, location privacy is not our concern, as fine-grained location information is definitely necessary for geo-social network providers to provide their services.

7 Conclusions

In this paper, we proposed a effective location verification scheme for "check-in" based geo-social networking providers, which provides check-in authenticity. The scheme uses features extracted from environmental WiFi signals to construct location tags, and measures the similarity of the location tags reported by the user device and venue device, to verify whether a user is indeed at the venue she claimed. Due to the dynamic nature of environmental signals, it becomes very hard for users to falsify location claims. In addition, the scheme does not require any change to the WiFi infrastructure. We also proposed an anonymity-based privacy-preserving check-in scheme for users, which can protect the temporal privacy of user check-ins. In the scheme, user use periodic pseudonyms which are generated using blind signature when checking-in at venues, to prevent providers tracking users. Partially blind signatures are used by the geo-social networking provider in generation of

check-in confirmation messages, to fight against the self-promotion attacks. Finally, a proof-of-concept implementation confirms the feasibility of our schemes in real-world mobile devices.

Acknowledgements

This work was supported by National Science Foundation of China (Grant No. 61373016) and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD). We would like to thank anonymous reviewers for their constructive comments and helpful advice.

References

[1] Qype. http://www.qype.com.

[2] Loopt. http://www.loopt.com.

[3] Facebook places. https://www.facebook.com/places/.

[4] Foursquare. https://foursquare.com/.

[5] W. He, X. Liu, M. Ren, Location Cheating: A Security Challenge to Location-based Social Network Services, *Proc. of the International Conference on Distributed Computing Systems (ICDCS)*, Minneapolis, Minnesota, 2011, pp. 740-749.

[6] S. Saroiu, A. Wolman, Enabling New Mobile Applications with Location Proofs, *Proc. of the Workshop on Mobile Computing Systems and Applications (HotMobile)*, Santa Cruz, California, 2009, pp. 3-8.

[7] W. Luo, U. Hengartner, Proving Your Location Without Giving Up Your Privacy, *Proc. of the Eleventh Workshop on Mobile Computing Systems and Applications (HotMobile)*, Annapolis, Maryland, 2010, pp. 7-12.

[8] Z. Zhu, G. Cao, Applaus: A Privacy-preserving Location Proof Updating System for Location-based Services, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Shanghai, China, 2011, pp. 1889-1897.

[9] M. Talasila, R. Curtmola, C. Borcea, Collaborative Bluetooth-based Location Authentication on Smart Phones, *Pervasive and Mobile Computing*, Vol. 17, No. PA, pp. 43-62, February, 2015.

[10] D. Chaum, Blind Signatures for Untraceable Payments, *Proc. of Advances in Cryptology (CRYPTO 82)*, Santa Barbara, California, 1982, pp. 199-203.

[11] D. Chaum, Blind Signature System, *Proc. of Advances in Cryptology (CRYPTO 83)*, Santa Barbara, California, 1983, pp. 153-153.

[12] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, D. Boneh, Location Privacy via Private Proximity Testing, *Proc. of Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, California, 2011, pp. 35-51.

[13] I. Polakis, S. Volanis, E. Athanasopoulos, E. P. Markatos, The Man Who Was There: Validating Check-ins in Location-based Services, *Proc. of the 29th Annual Computer Security Applications Conference (ACSAC)*, New Orleans, Louisiana, 2013, pp. 19-28.

[14] J. T. Chiang, J. J. Haas, Y.-C. Hu, Secure and Precise Location Verification Using Distance Bounding and Simultaneous Multilateration, *Proc. of the Second ACM Conference on Wireless Network Security (WiSEC)*, Zurich, Switzerland, 2009, pp. 181-192.

[15] D. Singelee, B. Preneel, Location Verification Using Secure Distance Bounding Protocols, *Proc. of IEEE International Conference on Mobile Adhoc and Sensor Systems Conference (MASS)*, Washington, DC, 2005, pp. 1-7.

[16] J. T. Biehl, A. J. Lee, G. Filby, M. Cooper, You're Where? Prove It!: Towards Trusted Indoor Location Estimation of Mobile Devices, *Proc. of 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'15)*, Osaka, Japan, 2015, pp. 909-919.

[17] M. Talasila, R. Curtmola, C. Borcea, Link: Location Verification through Immediate Neighbors Knowledge, *Proc. of Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous)*, Sydeny, Australia, 2010, pp. 210-223.

[18] R. Khan, S. Zawoad, M. M. Haque, R. Hasan, Who, When, and Where? Location Proof Assertion for Mobile Devices, *Proc. of Data and Applications Security and Privacy XXVIII (DBSec)*, Vienna, Austria, 2014, pp. 146-162.

[19] K. Vu, R. Zheng, J. Gao, Efficient Algorithms for K-anonymous Location Privacy in Participatory Sensing, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Orlando, Florida, 2012, pp. 2399-2407.

[20] X. Liu, K. Liu, L. Guo, X. Li, Y. Fang, A Game-theoretic Approach for Achieving K-anonymity in Location Based Services, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Turin, Italy, 2013, pp. 2985-2993.

[21] B. Niu, Q. Li, X. Zhu, G. Cao, H. Li, Achieving K-anonymity in Privacy Aware Location-based Services, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Toronto, ON, Canada, 2014, pp. 754-762.

[22] X. Zhao, L. Li, G. Xue, Secure Cloud-assisted Location Based Reminder, *Proc. of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security (ASIACCS)*, Hangzhou, China, 2013, pp. 323-328.

[23] X. Liu, H. Zhao, M. Pan, H. Yue, X. Li, Y. Fang, Traffic-aware Multiple Mix Zone Placement for Protecting Location Privacy, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Orlando, Florida, 2012, pp. 972-980.

[24] Y. Wang, D. Xu, X. He, C. Zhang, F. Li, B. Xu, L2p2: Location-aware Location Privacy Protection for Location-based Services, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Orlando, Florida, 2012, pp. 1996-2004.

[25] X. Li, T. Jung, Search Me If You Can: Privacy-preserving Location Query Service, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Turin, Italy, 2013, pp. 2760-2768.

[26] E. Novak, Q. Li, Near-pri: Private, Proximity based Location Sharing, *Proc. of IEEE International Conference on Computer*

Communications (INFOCOM), Toronto, Canada, 2014, pp. 37-45.

[27] J. Shao, R. Lu, X. Lin, FINE: A Fine-grained Privacy-preserving Location based Service Framework for Mobile Devices, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Toronto, Canada, 2014, pp. 244-252.

[28] M. E. Andres, N. E. Bordenabe, K. Chatzikokolakis, C. Palamidessi, Geo-indistinguishability: Differential Privacy for Location-based Systems, *Proc of ACM SIGSAC Conference on Computer & Communications security (CCS)*, Berlin, Germany, 2013, pp. 901-914.

[29] X. Chen, X. Wu, X. Li, Y. He, Y. Liu, Privacy-Preserving High-Quality Map Generation with Participatory Sensing, *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Toronto, Canada, 2014, pp. 2310-2318.

[30] M. Abe, E. Fujisaki, How to Date Blind Signatures, *Proc. of International Conference on the Theory and Applications of Cryptology and Information Security (ASIACRYPT)*, Kyongju, Korea, 1996, pp. 244-251.

[31] Z.-J. Zhang, X. Wang, L. Sun, Mobile Payment Anomaly Detection Mechanism based on Information Entropy, *IET Networks*, Vol. 5, No. 1, pp. 1-7, January, 2016.

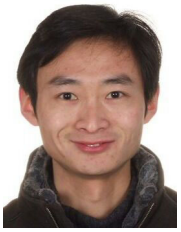
[32] Y. Ren, J. Shen, J. Wang, J. Han, S. Lee, Mutual Verifiable Provable Data Auditing in Public Cloud Storage, *Journal of Internet Technology*, Vol. 16, No. 2, pp. 317-323, March, 2015.

[33] Z. Fu, K. Ren, J. Shu, X. Sun, F. Huang, Enabling Personalized Search over Encrypted Outsourced Data with Efficiency Improvement, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No. 9, pp. 2546-2559, September, 2016.

[34] H.-L. Kim, J. G. Breslin, H.-C. Chao, L. Shu, Evolution of Social Networks based on Tagging Practices, *IEEE Transactions on Services Computing*, Vol. 6, No. 2, pp. 252-261, April-June, 2013.

Biographies

Yonglei Yao received his Ph.D. degree from Beijing University of Posts and Telecommunications in 2007. He is currently an associate professor at the School of Computer and Software, Nanjing University of Information Science & Technology. His research interests include privacy and security in wireless networks, Wearable Devices and pervasive computing.



Jiezhong Zhu received his master degree in computer science from Hohai University in 2004. He is currently an associate professor at Binjiang College, Nanjing University of Information Science & Technology. His research interests focus on security and privacy in wireless networks, Wearable Devices and pervasive computing.

Jingfa Liu received his Ph.D. degree in computer science from Huazhong University of Science and Technology in 2007. Now he is a professor in Nanjing University of Information Science & Technology. His research interests mainly include intelligent computation, algorithm design and optimization.

Neal N. Xiong received his both Ph.D. degrees in Wuhan University (about software engineering), and Japan Advanced Institute of Science and Technology (about dependable networks), respectively. Now he is a Professor at School of Computer Science, Colorado Technical University, Colorado Spring, CO, USA. His research interests include Cloud Computing, Security and Dependability, Parallel and Distributed Computing, Networks, and Optimization Theory.