Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 6031715, 12 pages
https://doi.org/10.1155/2018/6031715

Research Article

A Probabilistic Privacy Preserving Strategy for Word-of-Mouth Social Networks

Tao Jing, Qiancheng Chen, and Yingkun Wen

School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, China

Correspondence should be addressed to Qiancheng Chen; 16120048@bjtu.edu.cn

Received 3 May 2018; Accepted 21 June 2018; Published 8 July 2018

Academic Editor: Fuhong Lin

An online social network (OSN) is a platform that makes people communicate with friends, share messages, accelerate business, and enhance teamwork. In the OSN, privacy issues are increasingly concerned, especially in private message leaks in word-of-mouth. A user's privacy may be leaked out by acquaintances without user's consent. In this paper, an integrated system is designed to prevent this illegal privacy leak. In particular, we only use the method of space vector model to determine whether the user's private message is really leaked. Canary traps techniques are used to detect leakers. Then, we define a trust degree mechanism to evaluate trustworthiness of a communicator dynamically. Finally, we set up a new message publishing system to determine who can obtain the message of publisher. Secrecy performance analysis is provided to verify the effectiveness of the proposed message publishing system. Accordingly, a user in social networks can check whether other users are trustworthy before sending their private messages.

1. Introduction

Fog computing is a distributed collaborative architecture that enables specific applications or services between actual data sources and the cloud to be managed in the most efficient place [1]. This type of computing is effectively extending cloud computing capabilities and services to the edge of the network, bringing their advantages and functions to the point where data can be executed and manipulated in the closest proximity. In other words, fog computing is an extension of the concept of cloud computing. Different from cloud computing, fog computing wins by volume, emphasising quantity, no matter how weak a single computing node is. Cloud computing adjusts computing power, typically by a high performance computing device in a stack [2].

Fog computing-based typical services in online social networks (OSNs), such as Wechat, Facebook, Twitter, and LinkedIn, gradually become a primary mode to interact and communicate between participants. Each user in an OSN is a node component that makes up the social network topology. These nodes do not have strong computing and storage capabilities, but they can help with things like data transfer. Therefore, it is a reliable assumption to apply fog computing to social networks. Edge nodes in social networks are more mobile and more decentralized. The most concerned issue in social networks is privacy leaks. In the process of communication among these edge nodes, there is a novel way of privacy leakage through word-of-mouth. Word-of-mouth is a form of privacy disclosure on social networks. This kind of privacy leakage exists in a large number of real social networks but is seldom studied. For simplicity, we call it a word-of-mouth social network.

A word-of-mouth social network exists in the real human-centric world, which can pass messages from one person to another by oral communication and finally lead to the rapid spread of messages. Sometimes, the spread of private information may be peeped, misused, and taken illegally by other strangers [3]. As a result, it is important to prevent privacy disclosure caused by word-of-mouth. In fact, the disclosure degree of one user's privacy spreading is related to how to control others to access his data. In addition, it also depends on how much and what data the user wants to release. To tackle this issue, access control has been envisioned as a promising and effective approach to protect privacy of a person's account [4–6]. We intend to carefully

design a strategy among users in the word-of-mouth social network to share their data within a trusted user set.

Considering the human-centric network, our objective is to design an approach to achieve trustworthy word-of-mouth information release. This approach can trace the source of privacy disclosure and automatically adjust the trust degree of nodes in OSNs. Our exploration of this uncharted area needs to answer the following three challenges. First, how can we determine whether one user's privacy has been already disclosed and detect the leaker? Second, how to update social relationship of a user after detecting information disclosure? Finally, how to prevent acquaintances from disclosing privacy through the manner of word-of-mouth?

Aiming at the first challenges, we intend to use the vector space model (VSM) and the canary trap technology to achieve message disclosure detection. The authors of [7] present a mathematical expression of a VSM to measure the similarity of two sets. Then Li et al. improve the calculation method of the VSM in [8]. They apply semantic resources to reduce the dimensionality of feature items. Similarly, we apply the VSM in this article to calculate the similarity between the suspected leaked messages and the published ones to determine whether the messages is compromised. After determining message leakage, we employ the canary trap technology to trace the source of the leaked messages. In brief, different versions of sensitive data are sent to suspected leakers to find which version gets leaked [9].

Next, we introduce the trust degree of a user to ensure secure data sharing to cope with the second challenge. The trust mechanism has been widely studied to achieve privacy preserving in traditional OSNs. The authors of [10] investigate a recommendation belief based on a distributed trust management model for peer-to-peer networks. They quantify and evaluate the reliability of nodes and introduce a similarity function to construct the recommended credibility. In [11], the authors propose a method to check users' credibility before they enter a network. In addition, the authors in [12] design a secure recommendation system for mobile users to learn about potential friends opportunistically. Different from the existing works, we employ the dynamic trust degree to classify the recipients and to rank the sensitivity of publisher to information that needs to be published. The reason is that a user will unintentionally disclose privacy and later regret the behavior if the user is in an emotional state at the time of posting [13]. In order to avoid incorrect access to the sensitive-privacy information, the premise of access control for users is to classify the privacy information clearly.

Moreover, in order to prevent the privacy leakage, one user's (publisher's) privacy should only be shared with other ones (recipients) in a "correct" user set in OSNs [14–16]. The set, composed of numerous trusted recipients, can be updated based on the dynamic trust value that is a personal perception of a publisher to recipients. In this paper, we allow the publisher to rate recipients based on the value to determine whether they are trustworthy or not and dynamically adjust privacy settings of recipients by privacy disclosure. We design a systematic information publishing algorithm to reduce the leakage probability of private messages. In this case, we can reduce the risk of illegal spread of messages [17, 18].

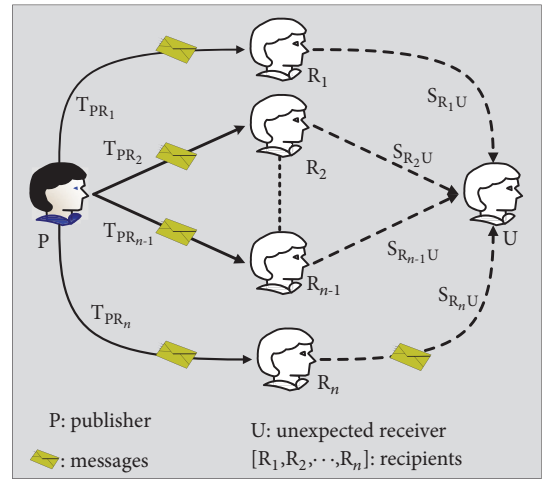

Figure 1: A privacy disclosure model in word-of-mouth OSNs.

To the best of our knowledge, there are few solutions to preserve privacy in word-of-mouth OSNs, which motivates our work. The main contributions of this paper are summarized as follows.

(i) We employ the VSM to measure the similarity between leaked messages and original ones so as to detect whether privacy disclosure exists. According to the tracing results by using the canary trap method, we determine which recipients are the possible leakers.

(ii) We define a dynamic approach to compute trust degree between users. To handle privacy leaks fairly, we design a novel utility function to punish the trust degree of leakers.

(iii) We design a probabilistic privacy preserving strategy based on a publisher's sensitivity to messages and recipients' trust degree for publishing a user's messages securely.

The rest of the paper is organized as follows. Section 2 introduces the privacy disclosure model in word-of-mouth OSNs. In Section 3, we present our privacy disclosure detection method via computing the similarity between leaked messages and original ones. Section 4 introduces the calculation method of trust degree and the punishment mechanism for leakers. Next, a trust degree based publishing system is proposed to achieve messages sharing with privacy preserving in Section 5. Then, we analyze the secure performance of our strategy in Section 6 and survey related work in Section 7. Finally, we conclude our work in Section 8.

2. System Model

In this section, we propose a privacy disclosure model as shown in Figure 1. This model can be formulated as a weighted directed social graph. Each node in the graph

denotes a user while a directed edge between two nodes represents the direction of messages transmission. We define T_{PR_i} and S_{R_iU} as the trust degree of the publisher P to its recipients R_i, $i \in [1 \cdots n]$, and the similarity between the unexpected receiver U with P's recipients, respectively. According to the graph of social relationship, we can extract the corresponding social attributes to explore a dynamic trust evaluation mechanism to achieve privacy preserving.

The privacy disclosure model in Figure 1 indicates that P is aware that its private messages are leaked to U after P sends the messages to R_i. The threat of privacy disclosure may be caused by one or more users in R_i. So the publisher is aware that his information is leaked and hoping to know which recipient sold him out. In this article, we hope to complete the tracking process of leakers. Because it is a post-evaluation method, we can only determine the probability that information will be leaked by a certain recipient. Recipients who have disclosed the privacy of the publisher should be penalized to protect the publishers' private message.

In this paper, we intend to employ the VSM algorithm to help A to determine whether the messages are really leaked out. Next, using the canary trap method to assist A to trace the possible leakers in the recipients set, and then give penalties to these leakers. (Here, what we consider is that recipients' motivation for disclosing information is not malicious. They are not trying to damage the publisher's interests. Therefore, the message disclosure should get the penalty of trust without considering legal or other punishment.) For the leakers, we dynamically adjust their trust degree to explore our privacy preserving strategy. Here, we present a novel utility function to measure the update standard of recipients' trust degree based on the centrality degree $Cen(i)$ of recipients and the similarity S_{ij} between two users. In our model, social networking platforms do not need to monitor messaging between users. The task of the platform is to help users find the leaker when they submit information detection applications.

3. Detection and Tracing Scheme for Leakers

A publisher realizes that his information may have been leaked, but he/she does not have enough confidence to determine what happened. In our model, the publisher can apply to the platform for leakage detection. The messaging platform uses the VSM to detect whether the publisher's information is actually leaked. In this section, we first compute the similarity score between suspected leaked messages and the corresponding published ones to determine whether the private message is leaked or not. The implementation of the VSM-based algorithm is actually quantifying the process by which publishers realize that privacy is compromised. In the second part, after privacy disclosure detection, we intend to employ the canary trap technology to trace privacy leakers.

3.1. Privacy Disclosure Detection. Comparing with images, files, and events, we believe that only texts leakage cannot be directly judged visually. Therefore, we only consider the private message to be a text and use the VSM method to calculate the similarity between suspected text and the original one.

In general, we can characterize text as a form of space vector based on the VSM. Therefore, the similarity between two texts can be measured by computing the similarity between two vectors. Assume that $(t_1, t_2, \ldots, t_n)$ indicate the text to be detected and $(w_1, w_2, \ldots, w_n)$ denote the corresponding coordinate values of the n-dimension space. Then, we intend to exploit the VSM to find out a score that indicates the degree of semantic equivalence between two texts. The following details describe the procedure to determine whether two texts are similar or not.

3.1.1. Text Preprocessing. First, we use the NLPIR word segmentation system to complete the word segmentation (NLPIR system is a software developed by the Institute of Computing Technology, Chinese Academy of Sciences; the principle of this system is based on the information cross entropy to automatically discover new language features and adapt to the language probability distribution model of the test corpus to realize adaptive participle), and obtain n word sets $(s_1, s_2, \ldots, s_n)$ that contain all the words that appear in the text. Next, we continue to remove the stop word that refers to the words with high frequency but no practical meaning. Such words include prepositions, adverbs, and conjunctions. They usually have no definite meaning in themselves, and only when they are placed in a complete sentence can they have a certain effect. The widespread use of a stop word in documents can easily cause interference with effective information. It is very significant to eliminate noise before feature weighting and selection. As a result, we remove all stop words in the word set s_j, $j \in [1 \cdots n]$.

3.1.2. Feature Extraction and Weight Calculation. As feature items, higher analytical accuracy of phrases and sentences may result in higher analysis error rate. In this paper, we choose words as a feature of text rather than phrases and sentences. In order to better reflect the performance difference of feature terms in the text content, we assign a weight value to each feature term. In particular, we calculate the weight of feature terms of each text segment separately. For the jth paragraph of the ith text s_{ij}, the feature term weight refers to the performance of the feature term t_k in the text segment s_{ij}. The formula for computing weights of feature terms can be indicated as follows:

$$w_{ijk} = tf_{ijk} = \frac{c_{ijk}}{l_{ij}}, \tag{1}$$

where tf_{ijk} is the frequency of the occurrence of feature term t_k in text segment s_{ij}, c_{ijk} denotes the number of feature terms t_k in the text segment s_{ij}, and l_{ij} represents the number of words contained in the text segment s_{ij}.

3.1.3. Text Vectorization. After extracting feature terms of each paragraph and assigning the corresponding weights, the text can be expressed as the form of a vector matrix. If the text D is divided into n parts and each part has m feature terms. The text D can be expressed as follows:

Input: text D
Output: the vectorization result $V[D]$ of text D
(1) $V[D] = [\overrightarrow{V_1}\overrightarrow{V_2}\cdots\overrightarrow{V_n}]^{-1}$
(2) $V_i = [x_1, x_2, \ldots, x_n]$
(3) Let section(D) be the segmentation result of the text D
(4) **For** each segmentation in section(D) **do**
(5) Word set $s_j \longleftarrow$NLPIR_segmentword(s_i);
(6) **For** each w_i in words set s_j **do**
(7) **While** stop words list contains w_i **do**
(8) Remove w_i from word set s_j
(9) T$\longleftarrow$extract features in s_j
(10) **For** each w_j in T **do**
(11) $m \longleftarrow$count(w_j)/countword(s_j)
(12) $V_i \longleftarrow mT$
(13) $V[D] \longleftarrow V[D] + V_i$
(14) Return $V[D]$

Algorithm 1: Vectorization procedure.

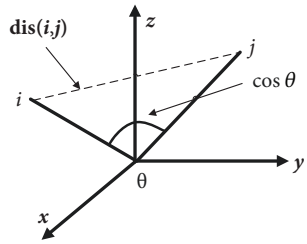

Figure 2: Cosine similarity and Euclidean distance in three dimensions.

$$D = [s_1, s_2, \ldots, s_n]^{-1} \times [t_1, t_2, \ldots, t_n] = \begin{pmatrix} w_{11} & \cdots & w_{1m} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nm} \end{pmatrix}. \quad (2)$$

The process of text vectorization is shown in Algorithm 1. Here, we segment paragraphs and words, remove the stop words, extract the feature terms, and calculate the weight of feature terms.

In Algorithm 1, "section(D)" denotes the processing procedure of text segmentation, "NLPIR_segmentword" represents a segmentation interface, "stop words list" represents the disabled word list defined in the text preprocessing part, "count" is a function that calculates the number of occurrences of each feature term in the text, and "countword" is a function that calculates the number of words contained in the text.

3.1.4. Similarity Measurement. Cosine similarity and Euclidean distance are the most common methods to measure similarity, as shown in Figure 2.

From Figure 2, we can see that cosine similarity measures the angle of vector space, which shows the difference of vectors in direction. Euclidean distance measures the absolute distance between two points. The cosine similarity is not sensitive to the absolute data, so it is applicable to the similarity analysis of specific features. Therefore, we take advantage of the cosine distance calculating method to measure the similarity. Intuitively, two vectors (texts) are independent or irrelevant if $Sim(i_1, i_2)$ is equal to 0, while $Sim(i_1, i_2)$ is equal to 1 when two vectors (texts) are the same. The similarity between information i_1 and information i_2 can be calculated as follows:

$$Sim(i_1, i_2) = \frac{1}{n}\sum_{j=1}^{n} \cos\theta_j = \frac{1}{n}\sum_{j=1}^{n} \frac{\overrightarrow{v_{1j}} \cdot \overrightarrow{v_{2j}}}{\|v_{1j}\| \cdot \|v_{2j}\|} = \frac{1}{n}\sum_{j=1}^{n} \frac{2\sum_{k=1}^{m_j} w_{1jk} w_{2jk}}{\sqrt{\left(\sum_{k=1}^{m_j} {w_{1jk}}^2\right)\left(\sum_{k=1}^{m_j} {w_{2jk}}^2\right)}}. \quad (3)$$

In order to verify the feasibility of our solution, we implement the algorithm using Python tools on Windows. Due to limited space, we only introduce the experimental text we selected and experimental results. We carried out vector extraction operations on three sentences shown in Table 1 and calculated the similarity between them. Finally, according to the cosine distance, we obtain the similarity between three texts, respectively, as shown in Table 1.

Table 1: The result of similarity calculation.

sentences	similarity scores
1. Julie loves me more than Linda loves me	(1,2) 0.753602532225
2. Jane likes me more than Julie loves me	(1,3) 0.128408027002
3. He likes basketball more than baseball	(2,3) 0.257423195662

We simply showed how to calculate the similarity between texts. When the publisher realized that their sensitive information had been compromised, OSNs calculate the similarity between suspicious information and target information, to determine whether publisher's information is actually compromised or not. After a lot of experiments, we concluded that when the similarity score is greater than 0.75, the meaning of the sentence is basically the same. Therefore, in this chapter, we say that the text-privacy has been leaked when the similarity score between the suspect text and the target text is greater than 0.75.

When a publisher realizes that their a sensitive message has been compromised, we would compute the similarity between a suspicious message I_s and the original message of publisher. In this case, we can determine whether publisher's privacy is actually compromised or not. The VSM approach quantifies the process of publishers' awareness of the leakage of private message. Once the publisher is aware of a privacy breach, they can adopt a canary trap approach to detect which users have recently leaked their private message.

3.2. Canary Trap Techniques. After conducting the VSM-based privacy disclosure detection, we assume that a publisher's privacy has been leaked. In order to detect leakers, it is necessary to have a strategy to determine whether or not such message is used by some user illegally. The canary trap is an

Table 2: Canary traps for different types of message.

Message types	Trapping settings
Digital images	Different watermarks
Database files	Different values of some cells
Events	Different values of some attributes
Texts	Mixture of different paragraphs

approach to detect information leakage source. The basic idea is that the publisher sends each suspect a different version of sensitive files and focuses on which version is leaked.

We consider different types of message using different traps for distinction according to the leakage message. We divide the types of message into four categories: digital images, database files, events, and texts as shown in Table 2. Accordingly, we provide four different types of message traps in our canary trap algorithm.

Recent events indicate that a user in publishers' friend circle leaks publisher's private message. After verifying the fact that private message is disclosed, we use the canary trap technique to find out leakers. First, we consider that the type of message trap is a digital image. We send an image embedded with different fingerprints to n users $R_1, R_2,...,R_n$. The digital fingerprint is embedded in each user's copy with a unique ID that can be extracted to help track the leaker when unauthorised leaks are found. We use digital watermarking to embed unique fingerprints in each copy of an image before releasing the image.

Definition 1 (digital watermarking). Digital watermarking technology directly inserts some identification information into the digital carrier and modifies the structure of a specific area without affecting the value of the original carrier. Digital watermarks are not easy to detect and remodify but can be identified by the manufacturer. Through these watermark information hidden in the carrier, we can achieve the purpose of confirming that the content creator and purchaser transmit secret information or determine whether the carrier has been tampered with.

For digital images represented by vector X, n recipients R_n is required, and the image owner generates unique fingerprint w_i for R_n. Watermark images will be passed to the recipient R_n that can be expressed as follows:

$$Y_i = X + JND \cdot w^i, \tag{4}$$

where Y_i is the digital image embedded with a fingerprint. JND is used to achieve the imperceptibility of embedding fingerprint in Y_i, which makes each copy unique. In this case, we can identify the leaker if dishonest users illegally repost its copy. We assume that recipients will not be able to decrypt the fingerprint by collusion, and a digital fingerprint identification system can track the leaker, as shown in Figure 3.

Secondly, for a database file, we modify data that is not important to implement different versions. What is not important here is that changing of its value does not affect the trend of the database. We need to generate n different versions of data. Supposing the number of data units that need to be changed is x, we can get $x = \log_2 n$. Similarly, once a database has been detected and leaked, we can judge the leaker.

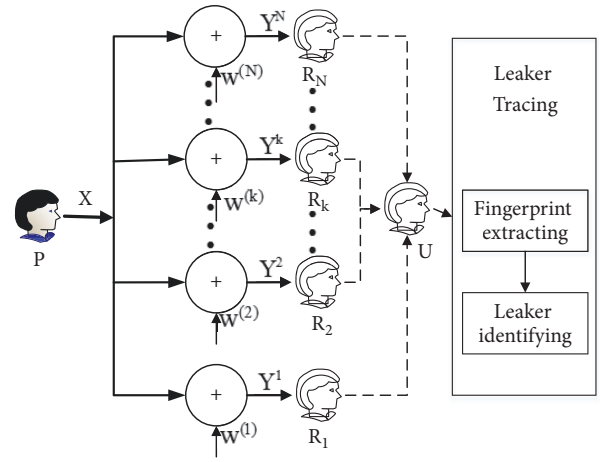

Figure 3: Digital fingerprinting system.

Thirdly, if we want to select a short event as an information trap, we modify the basic attributes like time, address, or target of the event. In short, n users $R_1, R_2,...,R_n$ may obtain n completely different times.

Finally, we define that a text has m paragraphs if we intend to select the text as an information trap. There are six different versions of each paragraph, and the mixing of these paragraphs is unique to each numbered copy of the text. Each version has minor changes, such as the font or spacing of words used in text. Unless someone tries to find the difference, it would not be noticed. There are more than 6^m possible permutations, but the actual text has only n numbered copies (assume $6^m > n$). If someone refers to two or three paragraphs of these paragraphs, we know which copy he sees, so we know who leaked private message.

We consider two special cases of false positive and false negative. First, the false positive condition means that the leaker in the last time privacy breach event might not have been detected by this canary trap experiment. But a person with a high frequency of leaks can not protect himself from every detection. Second, we are considering another possible scenario in which although a user accidentally leaked in the canary trap we deliberately falsified information. We cannot be absolutely sure that the previously detected sensitive information must be the user who leaked information in the canary trap. This is a false negative result. However, even if the previous message is not leaked by this user, he should still be punished for his mistake.

Although this approach can only determine the leaker of privacy in a certain probability, we still want to determine the person who leaks the privacy as fair as possible. What we need is to fairly assign different levels of trust penalties to different users. Therefore, we introduce the concept of similarity and centrality to distribute the trust degree penalty value fairly. This process will be described in detail in the next section.

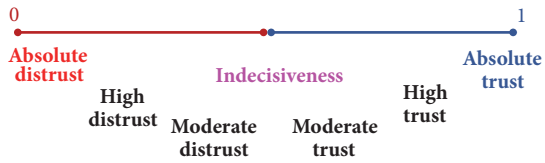

Figure 4: Trust degree range in an online social network.

4. Trust Degree Mechanism

In this section, we construct a trust degree mechanism to calculate and punish the trust degrees of the publisher to recipients. Trust acts as the glue that holds networks together, enabling networks to function effectively even though they lack a hierarchal power structure.

In the first part, we introduce a dynamic calculation approach of trust degree when there is no leakers. In the second part, we introduce how to punish the trust degree of leakers when there exist leakers.

4.1. Dynamic Calculation Approach of Trust Degree. In our mechanism, the value of trust degree ranges from 0 to 1. 0 represents absolute distrust while 1 means absolute trust; specific classification is shown in Figure 4.

The trust degree of a user to another user consists of two parts: direct trust degree (DT) and recommendation trust degree (RT). DT indicates the direct trust level of the publisher to the recipient based on the direct interaction experience. RT represents the recommendation trust and relies on rating information from the other recommender. We set P, R, and M to represent publisher, recipient, and recommender, respectively. The trust degree of the user P to the user R is defined as $T(P, R)$, which can be computed as follows:

$$T(P,R) = \alpha \times DT(P,R) + \beta \times RT(P,R), \quad (5)$$

where $DT(P, R)$ indicates the value of direct trust degree and $RT(P, R)$ indicates the value of recommendation trust degree. α and β are the trust degree regulation factors. Their values are related to the proportion that the publisher pays attention to DT and RT. Generally, we can set α and β according to the interaction frequency between P (or M) and R. Specifically, if P transacts with R more frequently than M, then we can give α a higher value and β a lower value, and vice versa. Therefore, we need to calculate DT and RT, respectively.

DT(P,R): DT between the publisher P and the recipient R can be calculated as follows.

$$DT(P,R) = \frac{\sum_{k=1}^{K} E_k(P,R) \times f_k \times w_k(P,R)}{\sum_{k=1}^{K} f_k}. \quad (6)$$

(i) We assume that the publisher P has conducted a total of K transactions in the past with the recipient R.

(ii) The evaluation value of the kth transaction is $E_k(P, R)$. $E_k(P, R)$ is provided by the user P and belongs to $[0, 1]$.

(iii) f_k is a time fading function and is defined as follow.

Definition 2 (time fading function). In order to improve the authenticity and dynamic adaptivity of direct trust, we consider that the interaction behavior of the past has attenuated the direct trust effect compared to the current interaction behavior. Specifically, compared with the Kth current transaction, the importance of the kth transaction in the past same transactions is depreciated. We define this attenuated function as time fading function that can be calculated as follows.

$$f_k = \delta^{K-k}, \quad 0 < \delta < 1, \ 1 \le k \le K \quad (7)$$

where $f_k = 1$ is previous interaction without attenuation and $f_1 = \delta^{K-1}$ is the first interaction with the largest attenuation.

(4) Last, weight of the interaction behavior of the kth transaction is represented as $w_k(P, R)$. The interaction behavior is defined by user P and can be divided into five grades according to the magnitude of behavior. We assign different weights to different interactions, which can distinguish the effect of different interactions on trust to a certain extent. Weights of each grade from big to small are 1, 0.8, 0.6, 0.4, 0.2. Therefore, the direct trust degree $DT(P, R)$ of the $(K + 1)$th interaction can be calculated by (6).

RT(P,R): RT of user P to user R is converged by P for the direct trust of the all recommendation users to R. Here, RT is a comprehensive evaluation for R by all users who have been interacted with R, which represents the overall credibility of R in social networks. So we define the value of RT of user P to user R as

$$RT(P,R) = \frac{\sum_{M \in G} DT(M,R) \times C_{PM}}{\sum_{M \in G} C_{PM}}, \quad C_{PM} \ge \Theta \quad (8)$$

where $RT(P, R)$ is the recommendation trust degree of recipients, C_{PM} is the credibility of P put to M, and G represents a collection of all trusted recommended users. Value of C_{PM} is equal to the direct trust $DT(P, M)$ of publisher P to recommender M. In this process of calculating recommendation trust, the publisher needs to provide a constant threshold Θ. The publisher P adopts recommendation information only when trustworthiness $C_{P,M}$ of the recommender is greater than Θ.

4.2. Trust Degree Punishment of Leakers. In the previous section, we detect private message leakers through the canary trap techniques. In this section, we punish these leakers by reducing the trust degrees of them. We punish trust degree of leakers based on a utility function. The utility function is integrated by the similarity between recipients and unexpected receiver with the centrality of the recipients. Therefore, we introduce two concepts: the similarity between two users and the centrality of a user.

Definition 3 (similarity). The similarity indicates the degree of separation between two users. It can be calculated by the amount of common friends in social networks. Sociologists have found that if two people have one or more friends, they have a greater chance of knowing and meeting each other.

Definition 4 (centrality). The centrality of a user is the index of the relative importance of quantized user in social networks. There are many manners to define the centrality of the user, such as betweenness centrality, closeness centrality, and degree centrality. Among these manners, we apply the direct centrality calculating approach that is defined as the number of other users in direct contact with the user to calculate similarity.

Accordingly, the centrality of user i can be expressed as follows.

$$Cen_i(\tau) = \frac{\sum_{k=1}^{N} d_{ik}(\tau)}{N}, \tag{9}$$

where $d_{ik}(\tau) = 1$ or $d_{ik}(\tau) = 0$ indicates whether there is a connection or not between the user i and the user k at time τ. N is the number of users in the network. The similarity between two users can be derived as follows.

$$S_{i,j}(\tau) = 1 + \left|F_i(\tau) \cap F_j(\tau)\right|, \tag{10}$$

where $F_i(\tau)(F_j(\tau))$ denotes the set of friends of user $i(j)$ at time τ. When we compute the utility function below, we convolve with the centrality and the similarity. If the similarity is equal to zero, the utility function makes no sense. However, when two users have no mutual friends, the similarity between them is equal to zero. So we add 1 in (10).

We apply the utility function as the standard metrics to punish the recipient who spread publisher's privacy information. If leakers who make mistakes in the canary trap are friends of the unexpected receiver U, we calculate the social similarity between leakers and the user U by comparing their friendship lists. We also calculate the social centrality of leakers. Note that social similarity and social centrality only reflect the characteristics of the network structure. In addition, we also need to consider the dynamics of social networks. Because of the dynamic change of nodes, the comprehensive utility should be a time-varying function. To address dynamic characteristics and avoid cumulative effects, we define the utility function as the convolution of similarity and centrality with a factor decaying as time. The total utility function value is the convolution of $Cen_R(\tau)$ with $S_{R,U}(\tau)$. $Cen_R(\tau)$ is the centrality of the recipient and $S_{R,U}(\tau)$ is the similarity between the recipient R and the unexpected receiver U.

$$\begin{aligned} Y_{R,U}(T) &= S_{R,U}(T) \otimes \frac{1 - Cen_R(\tau)}{T} \\ &= \int_{\tau=0}^{T} S_{R,U}(\tau) \cdot \frac{1 - Cen_R(\tau)}{T - \tau}, \end{aligned} \tag{11}$$

where the convolution operation provides a time-decaying description of all prior values of similarity and centrality. And $Y_{R,U}(T)$ is updated each time by accumulation of similarity and centrality when a leak event occurs.

In the previous section, we can get recipients who make mistakes in the canary trap. Now, we punish them by reducing their trust degree values. Specific penalties scale depends on the utility function value of a leaker l_i. The higher the degree of utility function, the more the value of trust degree is reduced. Above we have proved that the higher the utility function of the leaker, the greater the probability of leaking information. Therefore, we calculate the value of the trust penalty to a leaker based on the ratio of the utility function of the leaker and the sum of all utility function value. We define this specific measure of punishment as follows:

$$T_{sub}(l_i) = \frac{Y_{l_i,U}(T)}{\sum_{i=1}^{L} Y_{l_i,U}(T)} * T_{sub}, \tag{12}$$

where $T_{sub}(l_i)$ is a specific penalty in trust degree of leaker l_i and L is a collection of leakers who make mistakes in canary traps. T_{sub} is a total attenuation in trust degree caused by a leak event of privacy information, and this value is determined on the publisher. After attenuation, the trust degree of a leaker can be calculated as follows.

$$T(P, l_i)' = T(P, l_i) - T_{sub}(l_i). \tag{13}$$

The utility function is proportional to the similarity between the leaker and the unexpected receiver, and inversely proportional to the centrality of the leaker. A leaker with a high utility function will be penalized with a high trust value. But the total trust penalty will be equal to the publisher's expectation of a trust penalty.

$$\sum_{i=1}^{|L|} T_{sub}(l_i) = T_{sub}. \tag{14}$$

Users who make more mistakes in canary traps will be punished with more trust values. Users who receive a trust penalty are likely to fall into lower security classes (this concept is described in the next section). So recipients who leak users' privacy too often will be less likely to receive private information from publishers.

So far, we complete the calculation process and punishment process of the trust degree mechanism. In the first part, we propose a calculation approach of trust degree in the case of no leakers. In the calculation approach of trust degree, the interaction between users can help to enhance the trust between users. We comprehensively take into account the direct trust of the publisher to the recipient and the comprehensive evaluation of the recipient by other users around the social network. This approach of trust calculation takes into account the difference in degree of influence of interaction between different time periods. Every interaction between users changes the value of trust degree. The recommendation trust of users also directly changes the publisher's trust in the recipient. In the second part, we demonstrate how to punish the trust degree of leakers in the case of existing leakers. Compared with the traditional method of trust degree mechanism, we has a better real-time and dynamic trust degree mechanism.

5. Design of Message Publishing System

In this section, we design a message publishing system to reduce the risk that publishers' private message would be

Input: Publisher's weighted social graph;
Output: Publishing rules;
(1) **Phase I:** The publisher classifies the sensitivity of the message;
(2) Let I be the message that user want to be published;
(3) $S_I \longleftarrow sensitivity(I)$;
(4) Execute sensitivity classification after each publishing.
(5) **Phase II:** Divide the secure level of the recipient;
(6)Let R be the recipient;
(7) $SL(R) = X, X \in [1, 2, 3, 4, 5]$.
(8) **Phase III:** Determine whether to share message I with R or not;
(9) **if** $S_I > SL(R)$ **then**
(10) **Return:** Message I can be sent to recipient R;
(11)
(12) **else**
(13) **Return:** Message I can not be sent to recipient R;
(14)
(15) **end**

Algorithm 2: Rating scheme.

compromised. First, we propose a rating scheme to classify security level of recipients and sensitivity of publisher. Next, we introduce our message publishing system model formally and analyze the work flow of the system.

5.1. Rating Scheme. We propose a rating scheme based on the trust degree of a publisher to recipients and publisher's sensitivity to message. The rating scheme protects the private message of the publisher in OSNs.

Definition 5 (publisher sensitivity rating). Rating of the sensitivity of a publisher is based on the fact that different publishers have different sensitivities. Lower sensitivity of a publisher has relatively lower demand for privacy protection while higher sensitivity of a publisher requires a strong protection. We divide the sensitivity of the publisher on specific message into five levels, expressed as $S_i(S_i = 1, 2, 3, 4, 5)$. Level 5 is the lowest sensitivity and level 1 is the highest sensitivity. Since the publisher may have different sensitivity requirements for information at different time, it is necessary to perform sensitivity rating at each publish request.

Definition 6 (the secure level rating). A publisher needs to evaluate the trust degree of recipients that they want to interact with. According to the trust degree mechanism proposed in the previous section, the publisher can obtain the trust degree of all recipients including leakers. The secure level (SL) of recipients is divided into five levels according to the trust degree $T(P, R)'$. We quantify this trust degree rating process, and the secure level is upgraded from 1 to 5, as shown in Table 3.

In Algorithm 2, we obtain the publisher's request level for message sensitivity and classify the recipient's secure level. Considering message sensitivity and the secure level of a recipient, the proposed algorithm can help the publisher determine whether to share their private message with recipients or not..

Table 3: Security level of receivers in OSNs.

Security level	Trust level	Value of trust degree
1	Absolute trust	0.8 to 1
2	High trust	0.6 to 0.8
3	Indecisiveness	0.4 to 0.6
4	High distrust	0.2 to 0.4
5	Absolute distrust	0 to 0.2

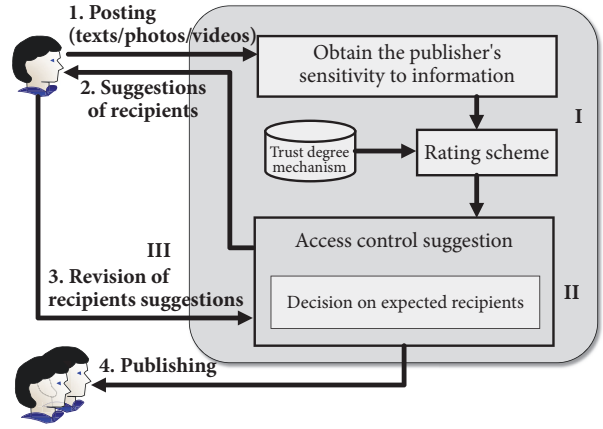

Figure 5: Overview of message publishing system.

5.2. System Overview. In order to prevent the leakage of privacy in OSNs, it is necessary to develop guidelines to determine the message publishing system. The system is exploited before a user posts on a social network site. The system determines who should be allowed to access messages and inform the user of the recommendation. In addition, users can add or exclude their own access to the proposed nonproprietary information.

The overview of the proposed system is shown as in Figure 5. It consists of three components: the rating scheme

module, the access control suggestion module, and the revision of recipient suggestion module. We describe these modules as below.

(i) **Rating scheme module:** A user needs to provide their sensitivity to the post while applying for publishing a post, because different users have different sensitivity to different messages. The system then queries the recipient's trust degree to build their secure levels. In this way, the system obtains the sensitivity level of the publisher to the messages and the secure level of the recipient. Then, the system can compare these values and formulate a access control suggestion for the next module.

(ii) **Access control suggestion module:** According to the level of disclosure determined in the previous step, this module determines the intended recipient based on the level of trustworthiness in the self-network of OSNs. We take into account the situation where the user is expected to send but the level of trust degree does not meet the requirements. We also take into account the situation where the level of trust degree meets the requirements but the user is not expected to send. During this procedure, it considers the revised messages associated with the previous posting and adds or excludes the intended recipient.

(iii) **Recipient suggestion revision module.** This module informs the publisher about the associated access control suggestion (analysis results). The access control suggestion contains a list of recipients. When a publisher gets a list of recipients, he/she can choose whether to publish information to the recipients in the list or revise the recipients in the list. The authorization/revision messages are stored in the system and would be considered in future access control recommendations.

The proposed message publishing system can help the publisher to choose the right recipients to access these messages. Because our system can prevent recipients who may leak privacy of the publisher from receiving messages, the risk of privacy leakage for the publisher can be reduced effectively.

6. Security Performance Analysis

In this section, we study the secure performance in terms of attack analysis, secrecy analysis, and access control analysis.

6.1. Attack Analysis. The leaking model we proposed indicates that recipients may spread privacy to other friends or strangers without user's consent. According to the privacy disclosure model in word-of-mouth OSNs shown in Figure 1, P shares its private messages to its recipients R_i, $i \in [1 \cdots n]$, but it is aware that the messages are known by U after a period of time. Users who have disclosed the private messages can probably be these recipients, and this manner of privacy disclosure is known as word-of-mouth.

By calculating the similarity between a suspicious message and the original one of publisher, we exploit the VSM to help P know whether its privacy is disclosed. Then, a canary trap technique is used to trace the source of disclosure. In this technique, we consider a possible scenario that we cannot be absolutely sure that the previously detected sensitive information must be the user who leaked in the canary trap, because the recipient leaked the information in this canary trap does not mean that he also leaked the previous privacy of the publisher. However, even if the previous messages are not leaked by the user, it should still be punished for its mistake. What we need is to fairly assign different levels of trust penalties to different users. As a result, we combine the centrality degree $Cen(R)$ of recipient R with the similarity S_{RU} between R and unexpected receiver U to obtain a utility function. The specific penalties scale depends on the utility function value of the leaker. Intuitively, the higher the degree of the utility function is, the more the value of trust degree is reduced. According to the dynamic evaluation of recipients' trustworthiness, decreasing of the recipient's trust degree directly affects the ability to obtain private messages. Finally, we set up a new message publishing system to determine who can obtain private messages.

6.2. Secrecy Analysis. According to the sensitivity degree of a publisher on private messages, our privacy preserving strategy first classifies users into different sensitive levels. Next, the publisher computes and classifies the trust degree of recipients. Finally, sensitivity rating and the secure level of the publisher are used to determine whether a recipient is allowed to obtain publisher's private messages or not. The trust-based privacy preserving strategy may change along with time, publisher itself, and other factors. In addition, the process may start by issuing an interactive request from the publisher to the recipients, so that the recipients cannot access a message without authorization.

6.3. Access Control Analysis. When a user in an online social network is ready to publish private messages, it considers that the private messages can be known only by a small group of its recipients rather than by random strangers. The proposed message publishing system can help the publisher to choose the right recipients to access these messages. Our system requires users to submit their own sensitivity levels to the post before applying for a post, because different users have different sensitivity to different messages. After the publisher sensitivity level acquisition process is complete, we count the secure level of the publisher's recipients through our dynamic trust degree mechanism and then give the publisher a list of suggested recipients of messages. The publisher can make revisions to the recipient list to delete or add and give feedback about the revision information to the access control system. This feedback also would be considered in future published access control recommendations. Because our system can prevent recipients who may leak privacy of the publisher from receiving messages, the risk of privacy leakage for the publisher can be reduced effectively.

7. Related Work

A service of online social networks (OSNs), such as Wechat, Facebook, Twitter, and LinkedIn, gradually becomes a

primary mode to interact and communicate between participants. A person can use these social applications at any time to contact with friends and send messages regardless of age, gender, and even socioeconomic status [19–21]. These messages should contain sensitive and private information, e.g., location [22], channel state information [23], routing information [24], social relationships [25], browsing data of Internet [26], health data [27], and financial transactions [28]. Obviously, the person should be worried about disclosures of personal information, which may be harmful to him either in virtual or real world [13, 29, 30]. As a result, an efficient privacy preserving strategy should be investigated to detect these threats [31].

The existing works in privacy preserving focus on information disclosure caused by malicious nodes (e.g., attackers or eavesdroppers) in OSNs [32, 33]. Essentially speaking, the authors of these works first convert nodes and the corresponding links in OSNs into vertices and edges of a weighted graph and then exploit graph theory and cryptography technologies to develop various protection solutions [34–36]. Although these solutions can protect personal information from network attacking and illegal eavesdropping, there still exists a more serious threat to users' privacy, i.e., information disclosure by word-of-mouth.

Text similarity calculation has been widely used in Internet search engine [37], intelligent question and answer, machine translation, information filtering, and information retrieval. In this paper, we use the text similarity calculation to determine whether the publisher's text information is leaked. In terms of algorithm, the most commonly used VSM in text similarity calculation was first proposed by Gerard Salton and McGill in 1969 [38]. The basic idea of the algorithm is to map the document to an n-dimensional vector, so as to transform the processing of text into a vector operation on a spatial vector. The similarity between documents is determined by comparing the relations between vectors. Among them, the most widely used weight calculation method is TF-IDF algorithm [39] and various improved algorithms. The most commonly used similarity measurement method is cosine similarity measurement [40].

To sum up, the privacy protection strategy in social network can be divided into two ways: role access control and data anonymity. The anonymous method is mainly used for multidimensional data such as network topology. For specific privacy such as user attributes, access control based on trust is a reliable way to protect privacy.

8. Conclusion

Considering privacy word-of-mouth disclosure by acquaintances, we put forward a novel privacy preserving strategy in this paper. In particular, we carefully combine the privacy leaking detection with the trust degree mechanism. The traditional privacy protection schemes are mainly on computing a trust degree threshold. In their schemes, a friend whose trust degree exceeds the threshold is considered believable. In contrast to these schemes, our strategy incorporates two new points of classifying each publisher's sensitivity and each ready interactive recipient. Our proposed publishing system consists of publisher's sensitivity level to information rating and recipient's secure level rating. What is noteworthy is that our scheme still gives the decision of message release to the publisher after giving the suggestion of access control. The publisher can make their own changes to the recipient list, and these revisions also will be considered in future published access control recommendations. Therefore, our information publishing strategy greatly reduces the risk of illegal spread of user's private messages.

In future, we want to provide a solution that is intended to prevent the recipient from illegally forwarding the message itself rather than the content of the message. Also, we may explore a sensitive message transmission interface for OSNs applications to protect users' privacy.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants nos. 61471028, 61571010, and 61572070) and the Fundamental Research Funds for the Central Universities (Grants nos. 2017JBM004 and 2016JBZ003).

References

[1] J. Barmagen, "Fog computing: introduction to a new cloud evolution," *Jos Francisco Fornis Casals*, pp. 111–126, 2013.

[2] H. R. Arkian, A. Diyanat, and A. Pourkhalili, "MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications," *Journal of Network and Computer Applications*, vol. 82, pp. 152–165, 2017.

[3] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, 2016.

[4] H. Yiliang, J. Di, and Y. Xiaoyuan, "The revocable attribute based encryption scheme for social networks," in *Proceedings of the International Symposium on Security and Privacy in Social Networks and Big Data, SocialSec 2015*, pp. 44–51, chn, November 2015.

[5] S. Machida, T. Kajiyama, S. Shigeru, and I. Echizen, "Analysis of Facebook Friends Using Disclosure Level," in *Proceedings of the 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, pp. 471–474, Kitakyushu, Japan, August 2014.

[6] S. Machida, T. Kajiyama, S. Shigeru, and I. Echizen, "Analysis of facebook friends using disclosure level," in *Proceedings of the 10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIH-MSP 2014*, pp. 471–474, jpn, August 2014.

[7] M. Suzuki, N. Yamagishi, T. Ishidat, M. Gotot, and S. Hirasawa, "On a new model for automatic text categorization based on

vector space model," in *Proceedings of the 2010 IEEE International Conference on Systems, Man and Cybernetics, SMC 2010*, pp. 3152–3159, tur, October 2010.

[8] L. Xu, S. Sun, and Q. Wang, "Text similarity algorithm based on semantic vector space model," in *Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pp. 1–4, Okayama, Japan, June 2016.

[9] F. P. Miller, A. F. Vandome, and J. Mcbrewster, "Canary Trap," in *1em plus 0.5em minus 0*, 4em Alphascript Publishing, 2010.

[10] J. Hu, Q. Wu, and B. Zhou, "RBTrust: A Recommendation Belief Based Distributed Trust Management Model for P2P Networks," in *Proceedings of the 2008 10th IEEE International Conference on High Performance Computing and Communications (HPCC)*, pp. 950–957, Dalian, China, September 2008.

[11] P. Yadav, S. Gupta, and S. Venkatesan, "Trust model for privacy in social networking using probabilistic determination," in *Proceedings of the 2014 4th International Conference on Recent Trends in Information Technology, ICRTIT 2014*, ind, April 2014.

[12] Y. He, F. Li, B. Niu, and J. Hua, "Achieving secure and accurate friend discovery based on friend-of-friend's recommendations," in *Proceedings of the 2016 IEEE International Conference on Communications, ICC 2016*, mys, May 2016.

[13] Y. Wang, G. Norcie, S. Komanduri, A. Acquisti, P. G. Leon, and L. F. Cranor, ""I regretted the minute I pressed share"," in *Proceedings of the the Seventh Symposium*, p. 1, Pittsburgh, Pennsylvania, July 2011.

[14] M. Gambhir, M. N. Doja, and Moinuddin, "Action-based trust computation algorithm for online social network," in *Proceedings of the 4th International Conference on Advanced Computing and Communication Technologies, ACCT 2014*, pp. 451–458, ind, February 2014.

[15] F. Nagle and L. Singh, "Can friends be trusted? Exploring privacy in online social networks," in *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining, ASONAM 2009*, pp. 312–315, grc, July 2009.

[16] Y. Yustiawan, W. Maharani, and A. A. Gozali, "Degree Centrality for Social Network with Opsahl Method," in *Proceedings of the 1st International Conference on Computer Science and Computational Intelligence, ICCSCI 2015*, pp. 419–426, idn, August 2015.

[17] A. Pandey, A. Irfan, K. Kumar, and S. Venkatesan, "Computing Privacy Risk and Trustworthiness of Users in SNSs," in *Proceedings of the 5th International Conference on Advances in Computing and Communications, ICACC 2015*, pp. 145–150, ind, September 2015.

[18] F. Riquelme and P. González-Cantergiani, "Measuring user influence on Twitter: a survey," *Information Processing & Management*, vol. 52, no. 5, pp. 949–975, 2016.

[19] Z.-J. M. Shen, "Integrated supply chain design models: a survey and future research directions," *Journal of Industrial and Management Optimization*, vol. 3, no. 1, pp. 1–27, 2007.

[20] A. M. Vegni and V. Loscrí, "A survey on vehicular social networks," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, article no. A3, pp. 2397–2419, 2015.

[21] J. Heidemann, M. Klier, and F. Probst, "Online social networks: a survey of a global phenomenon," *Computer Networks*, vol. 56, no. 18, pp. 3866–3878, 2012.

[22] Y. Huo, C. Hu, X. Qi, and T. Jing, "LoDPD: A Location Difference-Based Proximity Detection Protocol for Fog Computing," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1117–1124, 2017.

[23] L. Huang, X. Fan, Y. Huo, C. Hu, Y. Tian, and J. Qian, "A Novel Cooperative Jamming Scheme for Wireless Social Networks Without Known CSI," *IEEE Access*, vol. 5, pp. 26476–26486, 2017.

[24] M. Wang, J. Liu, J. Mao, H. Cheng, J. Chen, and C. Qi, "RouteGuardian: Constructing," *Tsinghua Science and Technology*, vol. 22, no. 4, pp. 400–412, 2017.

[25] X. Zheng, G. Luo, and Z. Cai, "A Fair Mechanism for Private Data Publication in Online Social Networks," *IEEE Transactions on Network Science and Engineering*, pp. 1-1.

[26] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, "Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity," *IEEE Access*, vol. 5, pp. 17020–17030, 2017.

[27] C. Hu, H. Li, Y. Huo, T. Xiang, and X. Liao, "Secure and Efficient Data Communication Protocol for Wireless Body Area Networks," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 2, no. 2, pp. 94–107, 2016.

[28] H. Alhazmi, S. S. Gokhale, and D. Doran, "Understanding social effects in online networks," in *Proceedings of the 2015 International Conference on Computing, Networking and Communications, ICNC 2015*, pp. 863–868, usa, February 2015.

[29] M. Fire, R. Goldschmidt, and Y. Elovici, "Online social networks: Threats and solutions," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2019–2036, 2014.

[30] M. Sleeper, J. Cranshaw, P. G. Kelley et al., ""I read my Twitter the next morning and was astonished" a conversational perspective on Twitter regrets," in *Proceedings of the 31st Annual CHI Conference on Human Factors in Computing Systems: Changing Perspectives, CHI 2013*, pp. 3277–3286, fra, May 2013.

[31] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *Proceedings of the IEEE INFOCOM*, pp. 1647–1655, April 2011.

[32] C. Akcora, B. Carminati, and E. Ferrari, "Privacy in social networks: How risky is your social graph?" in *Proceedings of the IEEE 28th International Conference on Data Engineering, ICDE 2012*, pp. 9–19, usa, April 2012.

[33] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, ICDE'08*, pp. 506–515, mex, April 2008.

[34] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving Privacy with Probabilistic Indistinguishability in Weighted Social Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1417–1429, 2017.

[35] L. Zhang, X.-Y. Li, K. Liu, T. Jung, and Y. Liu, "Message in a Sealed Bottle: Privacy Preserving Friending in Mobile Social Networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 9, pp. 1888–1902, 2015.

[36] R. Zhang, J. Zhang, Y. Zhang, J. Sun, and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 656–668, 2013.

[37] H. J. Zeng, Q. C. He, Z. Chen, W. Y. Ma, and J. Ma, "Learning to cluster web search results," in *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 210–217, July 2004.

[38] Salton and Buckley, "Team weighting approaches in automatic text retireval, readings in information retrieval," in *Proceedings of the in International Conference on Computer Vision Theory and Applications*, pp. 652–657, 1998.

[39] T. Peng, L. Liu, and W. Zuo, "PU text classification enhanced by term frequency-inverse document frequency-improved weighting," *Concurrency and Computation: Practice and Experience*, vol. 26, no. 3, pp. 728–741, 2014.

[40] V. Oleshchuk and A. Pedersen, "Ontology based semantic similarity comparison of documents," in *Proceedings of the 14th International Workshop on Database and Expert Systems Applications, DEXA 2003*, pp. 735–738, cze, September 2003.