A Trust Ranking Method to Prevent IM Spam

SUMMARY The problem of IM (Instant Messaging) SPAM, also known as SPIM, has become a challenge in recent years. The current anti-SPAM methods are not quite suitable for SPIM because of the differences in system infrastructures and characteristics between IM and email service. In order to effectively eliminate SPIM, we propose a trust ranking method in this paper. The mechanism to build up reputation network, global reputation and local trust ranking algorithms, reputation management, and SPIM filtering methods are presented. The experiments under five treat modes and algorithms enhancements are also introduced. The experiment shows that the proposed method is resilient to deal with SPIM attacks under several threat models.

key words: IM SPAM, SPIM prevention, trust ranking, threat model

1. Introduction

Instant Messaging (IM), one of the killer applications in the Internet, provides nearly real-time message delivery and presence information services. However, the problem of IM SPAM (also known as SPIM) has become a challenge since 2002. The record[1] said that 5%~8% of all enterprise IM today is SPIM, and IM users are estimated to get an average of 25 SPIM messages per day by 2008. Due to IM’s real time characteristic, SPIM may cause more severe damages than email SPAM.

Since SPIM and email SPAM have some common characters, some anti-SPIM solutions are based on traditional techniques to deal with email SPAM. However, they are not quite suitable for SPIM because of the differences in system infrastructures and characteristics between IM and email service[2].

In order to effectively eliminate SPIM, we present SpinRank: a SPIM prevention method based on global reputation and local trust ranking. This method integrates trust and reputation mechanisms with black-list filtering technique, and tracks users’ historical action records to deal with SPIM attacks in nearly real time, which is required in IM environment. Furthermore, SpinRank is resilient to deal with attacks under several threat models.

The paper is organized as follows: Section 2 discusses related work. In Sect. 3, we present a method to prevent SPIM called SpinRank, which is then evaluated under five threat models and enhanced in Sect. 4. Section 5 concludes the paper.

2. Related Work

Currently, SPIM prevention study is still in an early stage, such as black/white-list that uses privacy and contact-lists provided by IM service. Although the black/white-list methods are simple and effective, they are not the ultimate solutions because the constraint the applications of IM. In the business environment, there are always some new customers who are not on their white or black lists.

Ref. [2] provides an anti-SPIM method by integrating a number of mature SPAM defending techniques with modification and using their new techniques based on limiting IM sending rate. Such method could be deployed on both client and server/gateway to deal with SPIM attacks. In [2], authors evaluate every mature SPAM defending technique applied for IM service. Although SPAM and SPIM have some common characteristics, there is significant difference between them, especially the real time characteristic in IM service [2].

Trust and reputation mechanisms are popular in ranking a set of elements, such as web pages or people. Among the algorithms proposed for reputation scheme, the PageRank [3] algorithm is to compute Web-based reputation value. PageRank computes rank values by means of link analysis, i.e., based on the graph inferred from the link structure of the Web. The main idea of PageRank is that "a page has a high rank if the sum of the ranks of its back-links is high". Our trust ranking algorithm is inspired by this page ranking model. In the Web and P2P trust area, there are some related work [4][5][6][7]. Our local trust idea is also inspired by the related research work.

3. SPIMRANK

Compared with email service, the influence of SPIM attack is more serious to IM receivers than those encountering with email SPAM, because of the real time characteristic of IM service. Therefore, it’s necessary to detect and filter SPIM rapidly. In order to achieve such requirement, the design of filtering method should be effective and simple enough.

With guaranteed the authenticity of sender's identity, black/white-list techniques are simple and effective enough to filter SPAM (or SPIM). But as we have described above, such techniques may be not appropriate when dealing with
strangers that are neither in black nor white lists (we call them gray-list users). SpimRank will make use of trust and reputation mechanisms to deal with gray-list users when filtering incoming IM messages.

3.1 Design for Trust and Reputation Methods

The design for trust and reputation mechanisms includes such steps: (1) Reputation network build-up, (2) Trust and reputation algorithm design, and (3) Trust and reputation management.

3.1.1 Reputation Network Build-Up

In IM service, contact-list information are always stored at server side, which is convenient for server to collect them in order to form a social network. In SpimRank, we collect contact list information from IM server and build the reputation network. Such network can be shown as Fig.1: it is a directed graph (we call it $G_{white}$) where every node representing IM user, and the directed edge, i.e., $A \rightarrow B$ means that user A has added B into his contact list. Such network represents the good votes between users.

Furthermore, we can also form a reputation network based on blacklist filtering techniques, which is supported by most of current IM services. Figure 2 shows the blacklist-based reputation network (we call it $G_{black}$): the directed edge between node A and D means user A has added D into his blacklist. And such a network represents the bad (or punitive) votes between users.

From directed graph $G_{white}$, we define an $N \times N$ matrix $T_{white}$, where $N$ is the number of users in $G_{white}$.

Each element $T_{ij}$ in matrix $T_{white}$ is defined in Eq. (1), where $NumOfVote_i$ is the total count of users who have added i into their contact-list.

$$T_{ij} = \begin{cases} 1 & \text{if } j \text{ added } i \text{ into his contact-list} \\ 0 & \text{otherwise} \end{cases}$$  

We also define a vector $V_{black}$, where $V_{black}[i]$ equals the amount of users who have added $i$ into their blacklists (we call it Blacklist Level).

3.1.2 Trust and Reputation Algorithms Design

We calculate two kinds of reputation values: local trust value and global reputation value. When stranger C sends IM to A, C's local trust value, if exists, will be computed firstly to A. Otherwise, a global reputation value will be computed.

Algorithm for global reputation value ranking.

The basic idea of algorithm to compute global reputation value in SpimRank is based on PageRank: a user has a high rank if the sum of the ranks of users who add him into their contact-list is high. Equation (2) is borrowed from PageRank, we just substitute $T_{white}$ defined in Eq. (1) for the variable $T$ in the original equation of PageRank [3], to rank the global trust vector $GT$.

$$GT = c \cdot T_{white} + (1 - c) \cdot E$$  

The damping factor $c < 1$ (usually 0.85) is necessary to guarantee convergence and to limit the effect of rank sinks, which is a very simple attack studied on PageRank. The $E$ vector has all entries equal to $1/N$, where $N$ is the number of users in $G_{white}$. The main advantage of the algorithm is that they are designed for attack resilience.

In practice, we use power iteration algorithm to compute the approximate value of the vector $GT$ that represents the global reputation value.

Algorithm for local trust value ranking.

The calculation of the local trust value in SpimRank is based on such principles:

(1) Trust transitivity [9]. If A trusts B (means B exists in A's contact-list) and B trusts C, then we can confirm that A will also trust C on the recommendation from B.

(2) Compared with the user who has lower reputation value, users with higher reputation value will influence more heavily when rating to other user.

The algorithm to compute local trust value of C (relative to A):

(1) Find all reachable paths from A to C whose amount of hops less than 3 (such as path $A \rightarrow B \rightarrow C$ or $A \rightarrow B \rightarrow D \rightarrow C$, but path $A \rightarrow B \rightarrow D \rightarrow E \rightarrow C$ is invalid), named as $S_{path}$.

(2) For each path $p$ (from $A$ to $C$) in $S_{path}$, calculating
the trust value $E_p$ on this path:

$$E_p = \frac{\sum_{i \in p} w_i \times T_i}{|p|}$$  \hspace{1cm} (3)

In Eq. (3), $i$ is a node in path $p$, $|p|$ is the amount of node (exclude node $A$) in path $p$, $T_i$ is the global reputation value to node $i$, and weighted value $w_i = 2^{-\text{hop}(i, A)}$, where $\text{hop}(i, A)$ is the number of hop from node $i$ to $A$.

3. Calculating local trust value of $C$ (relative to $A$)

$$LT_{ac} = \frac{\sum_{p \subset S_{path}} E_p}{|S_{path}|}$$  \hspace{1cm} (4)

In Eq. (4), $p$ is a path in $S_{path}$ and $|S_{path}|$ is the amount of paths that $S_{path}$ contains.

3.1.3 Trust and Reputation Management

As far as trust and reputation management are concerned, we always discuss the architecture of the reputation system.

Most current public IM services, such as MSN, Yahoo!, do not cooperate with each other. In such autonomous IM service domains, the architecture of SpimRank reputation system can be deployed like Fig. 3: A Reputation Server Agent collect reputation information such as contact- and black-list from IM server, and compute, store, distribute the reputation value.

On the other hand, there are also many IM servers based on open IM protocols, such as XMPP protocol, etc, and it is familiar that these servers may cooperate with each other. In such multi-domain scenario, the architecture of our reputation system can be deployed like Fig. 4: a Register Server is built to manage registration and authentication of reputation server agents. What’s more, the register server also collects contact- and black-list information from those registered agents in order to form a global view of reputation network.

When a user’s reputation value is needed, the agents and Register Server will communicate with each other in the order shown in Fig. 5.

3.2 Design for SPIM Filtering Methods

The basic architecture of SpimRank filtering method can be described as Fig. 6.

As shown in Fig. 6, the incoming IM messages will be firstly filtered by black-list and white-list mechanisms.
When dealing with gray-list users, reputation-based filtering method will be used. In SpimRank, the reputation-based filtering method integrates trust and reputation mechanisms with blacklist technique. Furthermore, we define a new variable to record the amount of rejected IM messages that the sender gained, called: Num\_reject. The meaning of the rejected IM is the sender’s IM messages blocked by global (or local) blacklist or reputation-based filtering method. Such information can be stored as part of the user’s profile at server side and maintained as part of the state information when user logging on the server. By tracking such information at real-time, server can find out whose behavior is abnormal, i.e., whose Num\_reject is higher than normal level or increases faster, etc. After define such variable, we can define a SpimRank filtering method which is shown in Fig. 7.

The reputation-based filtering method described in Fig. 6 is shown in Table 1, which is represented by the function GetUserTypeByReputation(A).

As shown in Table 1, the function classifies users into two types: Spimmer and NormalUser. We collect user’s reputation value, Blacklist Level (V\_black), defined in Sect. 3 and Num\_reject information to determine his type. The variables Min\_rep, Min\_black and Min\_reject are threshold values that can be defined manually.

4. Experiments and Method Enhancement

4.1 Threat Models

To evaluate our method, we consider several strategies of malicious users to attack against SpimRank in order to send SPIM. Referred to the attacks on email system, there are five kinds of threat models:

**Threat Model A: Malicious Individual.**
Malicious peers in this model simply try to send SPIM to victims and do nothing else. This is the normal case.

**Threat Model B: Malicious Collective.**
Malicious peers in this model know each other and unite to form a “malicious collective”. In this collective, some of peers add other peers to their contact-lists in order to boost their reputation. Then the peers with high reputation will send SPIM to victims. On the other hand, some of the peers may impute some normal users by adding them into their blacklist, so that the Blacklist Level of these normal users will be increased.

**Threat Model C: One Time Identity Attack.**
Malicious peers always sending SPIM to victims like peers in threat model A did, but only use such address once.

**Threat Model D: Malicious Spies.**
Malicious peers in this model will use some normal users who are infected by virus or worms as zombie users, so that these victims may send SPIM or even boost these malicious peers to increase their reputation value.

**Threat Model E: Malicious Zombie User.**
Malicious peers in this model act like a normal user and get high reputation from normal social network. But they use their high reputation to boost other peers who send SPIM.

4.2 Experiment Method

Now we evaluate the effect of our filtering method in suppressing the treat models that described in Sect. 3.1. We simulate an IM network with 10,000 nodes, and define such variables shown in Table 2.

In order to get useful results and see the effect of our filtering method, we introduce our experiment method (call it AttackerCircleTest) in Table 3: for every user who participating in AttackerCircleTest, they act different behaviors depending on whether it is a normal user or not. The filtering method will be triggered when a spimmer sending SPIM to victims. After everyone has finished his action, we will remove the one who has been established as attacker. If all attackers we appointed have been removed or maximum
4.3.1 Threat Model A

Malicious peers will always sending SPIM to $N_{victims}$ victims. And victims will add such attackers to his blacklist in $P_{block}$ of all cases.

**Setup.** We simulate a network with such predefined parameters: $N_{friends} = 25$; $N_{attackers} = 100$. By altering parameters $N_{victims}$ and $P_{block}$, we execute AttackCircleTest method and get the results in Fig. 8: the X-axes represents $P_{block}$, and Y-axes represents ActualCircleCount. Every curve means $N_{victims}$ with different values.

**Discussion.** In Fig. 8, we can find that the larger $P_{block}$ is, the less ActualCircleCount is. If we assign a fixed value to $P_{block}$, i.e., 30, the ActualCircleCount value will be less in turn when $N_{victims}$ equals 20, 40, 60 respectively. It means that ActualCircleCount has relationship with both $P_{block}$ and $N_{victims}$.

Based on such conclusion, we do another experiment under $N_{friends} = 25$; $N_{attackers} = 100$, and results are shown in Fig. 9: We execute our AttackCircleTest method only once. After each time circulation, we will find out how many attackers’ Blacklist Level value has been exceeded the threshold $Min_{reject}$, defined as $N_{found}$. The X-axes represents $P_{block}$ multiply $N_{victims}$, Y-axes represents the ratio of amount of attackers we found with total amount of attackers. Every curve means parameter $Min_{reject}$ assigned to different value.

From experiment shown in Fig. 9, we can get the conclusion that when $N_{victims} \times P_{block} \geq 2 \times Min_{reject}$, the probability of finding out attackers can be raised to greater than 95% (i.e., when $Min_{reject} = 5$, $x = 2 \times Min_{reject} = 10$, we find that $y > 0.95$, similar result will be get when $Min_{reject} = 10$, 15, 20). It means that after this circle of AttackCircleTest, more than 95% of all attackers will be removed at the end of the next circle (because they have been added to the victims’ blacklist, and if they send SPIM to such victims again, their $Num_{reject}$ will be increased to exceed the threshold $Min_{reject}$).

As a result, the higher $N_{victims} \times P_{block}$ is, the faster we can find out the attacker. Let alone $N_{victims}$, it is very impor-
tant that the $P_{block}$ should be a high value in order to detect attackers quickly. [10] points that 86% of the email users simply delete email SPAM instead of reporting SPAM to the server. We can sure that, in IM service, the ratio of adding spimmer into its blacklist will be higher because the real-time characteristic of IM. Anyway, in order to increase the $P_{block}$ Value, we will provide a client side option allowing automatically adding SPIM attacker to user’s blacklist.

4.3.2 Threat Model B

Malicious peers form a malicious collective and boost some of the peers in their collective. Meanwhile, they also impute some normal users to increase their Blacklist Level, so that normal users may be blocked by our filtering method.

**Setup.** We simulate a network with pre-defined parameters: $N_{friends} = 25$; $P_{block} = 0.5$; $N_{variant} = 60$; $N_{attackers} = 100$. These attackers form a malicious collective and 30 of them are boosted by others in the collective. On the other hand, 50 of the normal users are imputed by peers in malicious collective and their Blacklist Level values are very high.

**Discussion.** We execute AttackerCircleTest experiment, and the results of Removed/Users is always 0. It’s because even though their Blacklist Level is high, our reputation-based filtering method described in Table 1 will also check the reputation and $Num_{reject}$ values after examine the Blacklist Level. And normal user always get a very low $Num_{reject}$ value and reputation value that larger than $Min_{rep}$. (Although the new user may get a low reputation value, it will cost much to find such new users and not the objectives for attackers to impute). On the other hand, the effect for attackers to boost peers in their collective is zero. It’s because the PageRank-based reputation algorithm is a flow model based algorithm [9]. Even though the malicious collective is large, it is isolated by the normal user’s community. Normal users will not informally add such malicious peers into their contact-list. As a result, the reputation flow will not transit into malicious collective and such boosting attack will be useless except that normal user’s community has links to the malicious collective. It may happen under the condition that there are malicious spies or zombie users in the normal user community. And we will discuss such scenario below.

Furthermore, we can find a serious problem that, the result of CurrentMissingSpinCount[1] always equals to $N_{variant} \times N_{attackers}$. It means that, at the first circulation, all SPIM are considered as normal IM messages by our filtering method and they are received by victims. The current algorithm of our filtering method is useless when attackers sending SPIM at the first time, and it may cause serious problem when dealing with “One time identity attack”. The solution to such flaw will be described in Threat Model C.

4.3.3 Threat Model C

**Setup.** We use experiments we execute above for reference and have conclusions that the current algorithm of filtering method has flaw when dealing with Threat Model C. We improve our algorithm which is shown in Fig. 10.

We add a new filtering approach FilteredByFriends before using reputation-based filtering method. Method FilteredByFriends can be described in Table 5.

**Discussion.** After improving our filtering method, we execute AttackerCircleTest under circumstance: $N_{friends} = 25$; $N_{attackers} = 100$; $P_{block} = 0.5$.

We change parameter $N_{variant}$ in every case and record experimental results: (1) RemovedAttackers[1] (2) $d = CurrentMissingSpinCount[1]/CurrentSpinCount[1]$.

The results we get can be shown in Figs. 11 and 12. We can find that the improvement has shown when dealing with Threat Model C, the more victim spinmer attacks, the more effective our filtering method take in the first circulation.

4.3.4 Threat Model D and E

Whoever the user is, either a malicious spy or virus-infected one, their behaviors are the same: they both act like a normal user, but add malicious peers into their contact-list to boost them. As a result, we combine the last two threat models into one experiment to find a effective way to detect such
BE: A TRUST RANKING METHOD TO PREVENT IM SPAM (SPIM)

**Fig. 11** Experiment result about RemovedAttackers after enhancement.

![Graph showing experiment result about RemovedAttackers after enhancement.](image)

**Fig. 12** Experiment result of $d$ after enhancement.

![Graph showing experiment result of $d$ after enhancement.](image)

attackers.

**Setup.** The current filtering method can only detect attackers, either a malicious peer or a virus-infected one, who sending SPIM to IM network directly. The ones who have normal user’s behavior not sending SPIM but boost other attackers can not be detected by our filtering method. We make further improvement in order to detect such malicious spies or zombie users. Figure 13 shows the enhancement to our filtering method.

When an attacker detected by our filtering method, and his $Num_{reject}$ is higher than our threshold $Max_{reject}$, the filtering approach $BackwardPunish$ shown in Table 6 will be triggered: the ones who add attacker into their contact-lists will be punished by increasing their $Num_{reject}$.

With our improvement, we execute the $AttackerCircleTest$ under the circumstance: $N_{friends} = 25$; $N_{attackers} = 150$; $P_{block} = 0.5$; $N_{victims} = (20, 50)$ and we randomly select 50 peers from normal user's community to be malicious spies (or can be seen as virus-infected users). Such spies randomly add attackers to their contact-list. The result of our experiments can be shown in Table 7.

**Discussion.** With our further improvement to the $SpimRank$ filtering method, the malicious spies and virus-infected users who boost to malicious peers can be detected effectively. We provide a backward-punish mechanism to punish users who boost to attackers.

5. Conclusions

With the fast deployment of IM applications, the SPIM attack has drawn serious public attention. Although there is similarity between SPIM and email SPAM, most of the anti-SPAM techniques can not be perfectly applied to SPIM prevention, because of the real-time characteristic of IM.
In this paper, we propose a new SPIM prevention method SpinRank, which is based on trust and reputation mechanisms, integrated with black-list and white-list techniques. We also make use of some unique mechanism in IM, such as maintaining session information for users logged on, and define a variable to record the user's historic action, which can be used to track user's abnormal behavior in order to detect SPIM attack as soon as possible.

The SpinRank has the following advantages:

(1) **Detecting SPIM attack in nearly real-time.** In SpinRank filtering method, we can upgrade the Blacklist Level and Name reject. Record immediately and servers always track such information in real-time. As a result, it is feasible to detect and defend attackers' abnormal behaviors as soon as possible.

(2) **Light-weight.** The filtering method makes full use of the IM characteristics, such as session information maintaining, contact-list and black-list mechanisms, etc. As a result, we do not need to change much to the original IM service. The design to filtering method is based on integrating trust and reputation mechanisms with black-list and white-list techniques, which is simple but highly effective.

(3) **High attack resilience.** The trust and reputation algorithm in SpinRank is based on a power iteration algorithm, which is typically highly resistant against attacks. And most of the threat models can be resisted. What's more, the filtering method can detect malicious spies and virus-infected users, which can't be detected by most of other methods.

(4) **Scalable computation.** Power iteration algorithms have been improved to be computationally feasible even for very large graphs [11].

In the future work, we intend to enhance the algorithms and move from a centralized reputation system to a distributed system to make the reputation system scalable for a large and multi-domains environment.

The prototype of SpinRank is implemented and is used in SureMsg system [12], an IM based security email system. Further exploration will be made on a testing environment in China Education and Research Network (CERNET).

**References**


**Jun Bi** received the B.S., M.S., and Ph.D. degree in computer science from Tsinghua University, Beijing, China. Currently he is a Full Professor and Director of Network Architecture & IPv6 Research Division, Network Research Center of Tsinghua University and Network Center of China Education and Research Network (CERNET). He was a postdoctoral scholar and research scientist of Bell Laboratories Research in New Jersey, USA. His research interests include Internet architecture and protocols, next generation Internet, high performance routers/switches, source address validation, Internet routing, IPv4/IPv6 transition, etc. He successfully led tens of government supported or international collaboration research projects, published more than 60 research papers, filed tens of innovation patents, and received national science and technology achievement awards. He serves as the members of editorial boards of several international journals and program committee chairs of tens of international conferences. He is the Senior Member of ACM, Senior Member of China Computer Federation, Senior Member of China Institute of Communications, Secretariat Director and Education Committee co-chair of Asia Future Internet Forum (AsiaFI).