International Journal of Automation and Computing 9(1), February 2012, 63-71
DOI: 10.1007/s11633-012-0617-5

METrust: A Mutual Evaluation-based Trust Model for P2P Networks

Chun-Ling Cheng Xiao-Long Xu Bing-Zhen Gao
College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, PRC

Abstract: It is necessary to construct an effective trust model to build trust relationship between peers in peer-to-peer (P2P) network and enhance the security and reliability of P2P systems. The current trust models only focus on the consumers′ evaluation to a transaction, which may be abused by malicious peers to exaggerate or slander the provider deliberately. In this paper, we propose a novel trust model based on mutual evaluation, called METrust, to suppress the peers′ malicious behavior, such as dishonest evaluation and strategic attack. METrust considers the factors including mutual evaluation, similarity risk, time window, incentive, and punishment mechanism. The trust value is composed of the direct trust value and the recommendation trust value. In order to inhibit dishonest evaluation, both participants should give evaluation information based on peers′ own experiences about the transaction while computing the direct trust value. In view of this, the mutual evaluation consistency factor and its time decay function are proposed. Besides, to reduce the risk of computing the recommendation trust based on the recommendations of friend peers, the similarity risk is introduced to measure the uncertainty of the similarity computing, while similarity is used to measure credibility. The experimental results show that METrust is effective, and it has advantages in the inhibition of the various malicious behaviors.

Keywords: Peer-to-peer (P2P) network, reputation, trust model, mutual evaluation, similarity risk.

1 Introduction

In peer-to-peer (P2P) network, peers have equal functionality and can communicate directly to exchange information and collaborate with each other. It provides a flexible and extensible computing environment for large-scale resources sharing, instant communication, cooperative working, etc. The nature of distributed, self-organized, anonymous and dynamic P2P network offers enormous opportunities and presents potential threats and risks too. There are a lot of selfish, deceptive, and malicious behavior in P2P networks[1]. It is hard to solve these problems efficiently by conventional security policies, such as authentication[2], authorization, confidentiality of communications[3], etc. As a result, trust mechanism in social network is introduced to improve the scalability and robustness of the P2P systems. Trust model collects, distributes, and aggregates feedbacks about participants′ past behavior to help peers to decide whom to trust.

Like the interpersonal relationships in social network, there are two kinds of trust between peers in P2P network: direct trust and recommendation trust. Direct trust means two peers directly exchange information, and the direct reputation is based on their direct experience, which usually is the consumer′s one-way evaluation to the transaction in existing trust models. Dishonest evaluation may be provided by some malicious peers to slander good peers or make strategic attacks. Recommendation trust is that two peers never exchange information directly, and they establish trust relationship based on recommendation from other peers. The recommendation reputation is from other peers′ evaluation, and its credibility depends on the credibility of recommendation information. In order to suppress misbehavior of malicious peers, such as slander and strategic attack, and to reduce the risk of recommendation trust, this paper presents a novel trust model based on mutual evaluation, called METrust. In order to suppress false evaluation, we propose the mutual evaluation consistency factor and its time decay function to give different weight to the consumer′s evaluation in direct trust computing. Moreover, to reduce the risk of recommending, the similarity risk is proposed to measure the uncertainty of the similarity of recommendation in recommending trust computing. Incentive and punishment mechanism is also adopted to improve the initiative of peers, encourage good behavior, and punish malicious peers.

The rest of this paper is organized as follows: First, in Section 2, we present a summary of related work on this topic. Second, in Section 3, we describe our trust model METrust in detail. Third, experiments and results are presented in Section 4. Finally, we draw the conclusions in Section 5.

2 Related work

Many trust models have been proposed for P2P networks and can be divided into global trust models and local trust models. In global trust models, each peer has a unique global trust value iteratively calculated by gathering the neighbor peers′ satisfaction, as described in some famous models, including EigenTrust[4], PowerTrust[5], and GossipTrust[6]. In local trust models, such as PeerTrust[7] and TrustGuard[8], a peer gets other peers′ trust value by querying some peers providing recommendation.

In recent years, some trust models are put forward

Manuscript received September 10, 2010; revised May 25, 2011
This work was supported by National Natural Science Foundation of China (No. 60873231), Research Fund for the Doctoral Program of Higher Education (No. 20093223120001), Science and Technology Support Program of Jiangsu Province (No. BE2009158), Natural Science Fund of Higher Education of Jiangsu Province (No. 09KJB520010) and Special Fund for Fast Sharing of Science Paper in Net Era by CSTD (No. 2009117).

to minimize the selfish and malicious peers′ behavior, which mainly include trust models based on feedback, dynamic time-window, recommendation, and some others. FCTrust[9] is a global trust model that distinguishes the feedback credibility from service credibility. FCTrust is based on feedback credibility (FC) to quantify and evaluate the trustworthiness of participants. TW-Trust[10] is a local trust model based on time-window. It considers the experience and recommendation′s time-sensitivity while computing the trust value of peer. As for recommendation based trust model, Tian et al.[11] proposed RETM, which is based on recommendation evidence and filters out noisy recommendation information before combining the evidences. Huang et al.[12] proposed a recommendation trust model based on Dempster-Shafer (D-S) evidence theory. It deduces a peer′s local trust values from its transaction history and then combines the peer′s local trust values by D-S combination rule and gets its global trust value finally. Another recommendation evidential trust model is based on the Dezert-Smarandache theory[13], which has a higher expressiveness than the trust models based on the D-S theory. Among others, an alternative social-network-based reputation ranking algorithm called Poisonedwater is proposed to solve front peers attacks[14]. Tian et al.[15] proposed a super-peer based trust model, in which peers gather in a group according to their interest similarity and unfair feedbacks are filtered by algorithm based on peers′ similarity. In addition, a reputation management algorithm for distributed hash table (DHT)-based P2P environment is presented to select "good" peers cooperating and withstand malicious activity of single malevolent peers and their collusions as well[16]. Moreover, a role-based trust model is presented to establish trust among anonymous peers exactly[17]. It builds credential graphs, which are converted into credential tree by depth-first search (DFS) algorithm for trust computation and delegation. These trust models enhance the robustness by different methods.

In the trust models mentioned above, only the consumer evaluates the transaction, but the provider′s evaluation is neglected. Therefore, the consumer may slander the server maliciously to collude or attack strategically. To evaluate the trustworthiness of participating peers exactly, a trust model based on mutual evaluation called METrust is proposed in this paper. METrust is constructed on trust overlay network (TON)[5, 18]. Trust computing relies on direct trust computing and aggregating the available feedbacks in the network in hope of achieving as much robustness as possible. In the direct trust computing, mutual evaluation and time decay function are used to compute the direct trust value objectively. In recommending trust computing, similarity risk that measures the uncertainty of the similarity computing is used to improve calculation accuracy. The trust value is stored in distributed way, and the incentive and punishment mechanism is adopted to update trust values. At last, the experimental results show that METrust outstands EigenTrust in both accurate trust computing and the inhibition of the malicious peers.

3 Trust model: METrust

3.1 Architecture of METrust

METrust is based on TON and each peer has two major modules: trust storage module and trust computing module. The distributed architecture of METrust is shown in Fig. 1.

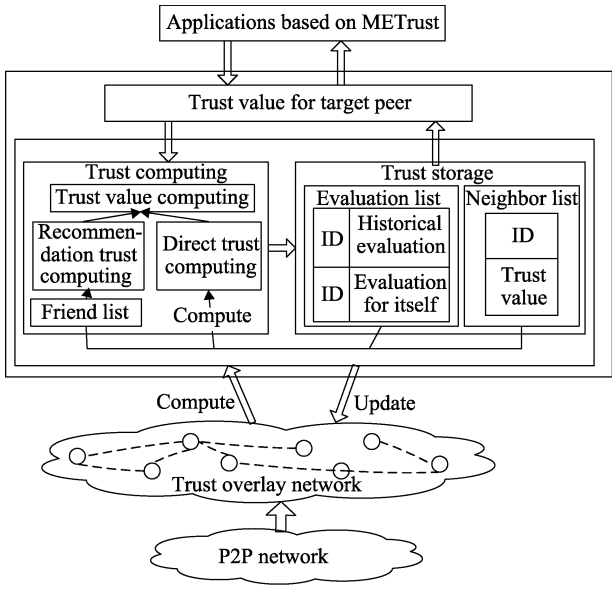


Fig. 1 The architecture of METrust

TON is established by extending topology adaptation algorithm proposed by Niu et al.[18] by which peers with higher reputation have higher probabilities of being accepted as neighbors. Therefore, METrust optimizes neighbor sets of peers. In addition, TON provides four important classes to manage connections, resources, neighbor peers, and transactions, respectively. These classes call the routing and peers management procedures supported by underlying P2P overlay network to achieve their functions.

To store the necessary information for trust computation, trust storage module maintains two lists: an evaluation list and a neighbor list. Evaluation list is used to preserve historical n times evaluations about direct transactions, including evaluations of remote peers and its own. N is controlled by time window. The smaller the time window is, the less the storage space is needed, but the less accurately the trust value is computed. Therefore, the time window is decided by specific applications. Neighbor list records the trust values of logical neighbor peers. Supposing that peer A downloads a file from peer B, the evaluation information is added to evaluation list. Then, the trust value of peer B is updated due to the evaluation of this transaction.

Trust computing module is responsible for calculating the trust value of the service provider. It consists of direct trust computing submodule and recommendation trust computing submodule. The direct trust computing submodule calculates the direct trust value by using mutual evaluation information in evaluation list on both consumer peer and provider peer. The recommendation trust computing submodule aggregates the recommendation information coming

from peers in friend list to get recommendation trust value. At last, trust computing module combines the direct and the recommendation trust value.

3.2 Deployment of METrust

When a transaction request is issued by peer A, class connection on peer A tries to connect other peers according to routing algorithm. Class connection on the peer, which receives the request, forwards it and meanwhile checks if it is able to respond to it. Then, class connection on peer A selects a peer with the highest trust value from responding peers and connects it. Supposing peer B is selected, with class transaction, peer A and peer B cooperate to carry out this transaction, get the mutual evaluation information from respective cache, and save it in the evaluation list.

When a transaction is completed, trust computing module on peer A checks the evaluation list to search peer A′s historical evaluation of peer B, and trust computing module on peer B reads the evaluation list in peer B′s trust storage to search the evaluation of its own. With the help of the evaluation information from two sides, the direct trust value of peer B will be computed by peer A using trust computing module. Then, peer A obtains its friend list. Subsequently, peer A aggregates the recommending trust value of peer B by using recommendation information given by peers in friend list. Finally, peer A calculates the trust value of peer B through the direct trust value and the recommendation trust value.

Friend list records the recommending peers in recommendation trust computation, and it is obtained dynamically. Recommending peer is the peer that has direct transaction with target peer (here it is peer B), and its reputation is higher than a threshold. Friend list is obtained by retrieving the neighbor list and the neighbor peer′s evaluation list iteratively. When peer A needs to compute the recommendation trust value of peer B, peer A retrieves its neighbor list and connects its neighbor peers. The neighbor peer searches its evaluation list to check whether it has ever interacted with peer B directly. If this is true and the reputation of this neighbor peer is higher than a threshold, this neighbor peer would be regarded as a friend peer of peer A. This process goes on until enough friend peers are found or time to live (TTL) is zero. In initialization, the friend peers are randomly selected because both lists are empty.

After a transaction, neighbor list is updated by topology adaptation algorithm. Peer A chooses some new peers as preneighbors from peer B′s evaluation list and neighbor list, and sends the requests to these peers for neighbor relationship. When receiving the requests, the preneighbor peers check the trust value of the requester. If it is higher than a threshold, both peers record each other in their neighbor lists, and the neighbor relationship is established. Thus, some peers with higher trust values are connected, which improves the quality of service, isolates malicious peers and resists collusion attack effectively. In initialization, the neighbor list is empty.

3.3 Trust value computing

In METrust, we suppose the trust value is distributed in the interval of [0, 1]. 1 means full trustworthiness, while 0 is definitely untrustworthy. For a new peer, we know nothing about it, and there is no way for us to determine whether it is good or bad. Therefore, we choose 0.5 as its initial trust value, which is the mean value on the close interval [0, 1]. The trust value is aggregated by the direct trust value and the recommendation trust values, which is calculated as shown in (1):

$$T_{ij} = \delta D_{ij} + (1 - \delta) R_{ij}. \tag{1}$$

T_{ij} is the trust value of peer i for target peer j. D_{ij} represents the direct trust value of j computed by i according to past n direct transactions. R_{ij} is the recommendation trust value of peer i for target peer j calculated by the recommendation information from friend peers of peer i. The direct trust factor $\delta\,(0 \leqslant \delta \leqslant 1)$ is peer i's confidence about its transaction-based rating for peer j. The larger δ is, the more possible i trusts itself. If peer i only trusts its direct transactions and does not care about the recommendations from other peers, δ can be 1. Otherwise, δ can be 0.

1) Direct trust computing

In most trust models, the direct trust value is computed according to the rating of the consumer to the provider. This rating may be given casually or even maliciously. In order to suppress dishonest evaluation, service consumer also reports the rating to itself after a transaction. If two ratings are opposite, that is, one is good, and the other is bad; there may be dishonest one. The effect of this evaluation should be decreased. Otherwise, if two ratings are consistent, the effect of this evaluation should be enlarged. Therefore, we propose the mutual evaluation consistency factor to give different weight to the consumer′s evaluation.

Once a transaction is completed, both participants will give evaluations. The evaluations of the transaction that occurred at different times will have different effects on the direct trust value. The nearer the time of transaction is, the more important the evaluation is. Therefore, in the computation of the direct trust, the time decay function is adopted. METrust uses time window whose length is n to save the latest n times of transaction evaluations. Supposing $[t1, t2, \cdots, tn]$ is a time window and represents the time of the latest n transactions between peer i and j, $d_{ij} = \{d_{ij}^{t_1}, d_{ij}^{t_2}, d_{ij}^{t_3}, \cdots, d_{ij}^{t_n}\}$ is the evaluations of peer i to j in time window above, in which $d_{ij}^{t_k}$ represents the evaluation of the service at t_k, and $d_{ij}^{t_n}$ represents the most nearest evaluation. $f(t_k)$ is time decay function, D_{ij} is the direct trust value peer i calculated for target peer j, as shown in (2).

$$D_{ij} = \frac{\sum_{t_k=t_1}^{t_n} f(t_k) d_{ij}^{t_k}}{\sum_{t_k=t_1}^{t_n} f(t_k)}. \tag{2}$$

Definition 1. Time decay function $f(t_k)$. The evaluation weight of transaction at t_k is defined as time decay function $f(t_k)$ detailed in (3). θ_{ij} is distributed in the interval of [0, 1], which makes nearer transactions have higher weight.

$$f(t_k) = \theta_{ij}^{t_n - t_k},\ 0 < \theta_{ij} < 1,\ t_1 \leqslant t_k \leqslant t_n. \tag{3}$$

From (3), we can conclude that the larger θ_{ij} is, the larger D_{ij} is. However, θ_{ij} is chosen subjectively and randomly in many proposed trust models, which destroys the objectivity of trust computation. We propose a novel method in which mutual evaluation consistency factor in Definition 2 is used to calculate $f(t_k)$. Both service consumer and provider report ratings about the outcome after a transaction. If the two ratings are consistent, θ_{ij} should be enlarged to make the direct trust of service provider increase, while if they are inconsistent, θ_{ij} should be lowered to decrease its direct trust value.

Definition 2. Mutual evaluation consistency factor θ_{ij} is the consistency degree of recent n transactions between peer i and j, as shown in (4).

$$\theta_{ij} = \frac{\sum\limits_{t_k=t_1}^{t_n} (d_{ij}^{t_k} - \overline{d_{ij}}) \times (d_{jj}^{t_k} - \overline{d_{jj}})}{\sqrt{\sum\limits_{t_k=t_1}^{t_n} (d_{ij}^{t_k} - \overline{d_{ij}})^2} \times \sqrt{\sum\limits_{t_k=t_1}^{t_n} (d_{jj}^{t_k} - \overline{d_{jj}})^2}} \tag{4}$$

where $d_{jj} = \{d_{jj}^{t_1}, d_{jj}^{t_2}, d_{jj}^{t_3}, \cdots, d_{jj}^{t_n}\}$ presents the ratings of peer j for its services in time window $[t_1, t_2, \cdots, t_n]$, $\overline{d_{ij}}$ is the average rating of recent n transactions that peer i calculates for server peer j, and $\overline{d_{jj}}$ is the average rating of peer j for its n services.

2) Recommendation trust computing

The recommendation trust of peer i toward peer j is aggregated by using friend peers′ recommending information. According to the trust level of peer i to its friend peer, peer i will give different weight to the recommendation. There is uncertainty about weight selection. We introduce the risk factor to measure the uncertainty.

Let Q be a group of friend peers of peer i toward peer j. f_{kj} is the rating of friend peer k for peer j. w_k denotes the credibility of the feedback submitted by k. Recommending trust is calculated as follows:

$$R_{ij} = \sum_{k \in Q} w_k \times f_{kj}. \tag{5}$$

As can be seen from (5), w_k is a kind of weight of recommending peer k. The larger w_k is, the more credible the recommendation of peer k is. w_k can be computed by measuring the similarity between peer k and other peers. METrust introduces cosine similarity measure. Let P_i denote the set of peers that have interacted with peer i, and P_k denotes the set of peers that have interacted with peer k. $P = P_i \cap P_k$, denotes the common set of peers, which have interacted with both peer i and peer k. In addition, n_P is the number of peers in P. The direct trust by peer i and the direct trust by peer k over P are modeled as two vectors, which are $(D_{i1}, D_{i2}, \cdots, D_{in_P})$ and $(D_{k1}, D_{k2}, \cdots, D_{kn_p})$. In some trust models[7], the similarity C_{ik} is defined in (6), and the credibility w_k can be calculated by C_{ik}, which is calculated in (7). W_{default} denotes the default value when P is empty.

$$C_{ik} = \cos\theta_{ik} = \frac{\sum\limits_{t=1}^{n_P} D_{it} \times D_{kt}}{\sqrt{\sum\limits_{t=1}^{n_P} D_{it}^2} \times \sqrt{\sum\limits_{t=1}^{n_P} D_{kt}^2}} \tag{6}$$

$$w_k = \begin{cases} \dfrac{C_{ik}}{\sum\limits_{t \in P} C_{it}}, & n_p \neq 0 \\ w_{\text{default}}, & n_p = 0. \end{cases} \tag{7}$$

However, the precision of the similarity C_{ik} depends on n_P. The less n_P is, the less precise the similarity is. Therefore, METrust introduces a risk factor CR_{ik} to measure the uncertainty in similarity computation.

Definition 3. Similarity risk factor CR_{ik} is a degree reflecting how peer i and k care about the risk in similarity computation.

Supposing that n_i denotes the number of peers that have interacted with peer i and n_k denotes the number of peers that have interacted with peer k, CR_{ik} is computed in (8). Moreover, w_k is calculated in (9) with the help of CR_{ik}, where $\alpha \in [0, 1]$. α is used to adjust the proportion of CR_{ik} in w_k. The smaller α is, the less it is concerned about similarity risk.

$$CR_{ik} = 1 - \frac{n_p}{n_i + n_k - n_p} \tag{8}$$

$$w_k = \begin{cases} \dfrac{C_{ik}}{\dfrac{\alpha CR_{ik}}{\sum\limits_{t \in P} C_{it}}}, & n_p \neq 0 \\ \dfrac{\alpha CR_{it}}{w_{\text{default}}}, & n_p = 0. \end{cases} \tag{9}$$

3) Incentive and punishment mechanism

There are many uncooperative peers in P2P systems, such as free riders, which only enjoy service from other peers, but do not offer service. Besides, some malicious peers do not give other peers feedbacks or even give false feedbacks. Incentive and punishment mechanism in METrust is implemented as follows: First, TON is constructed, by which peers with higher trust values are selected as neighbors, and malicious peers are isolated because of their low trust values. Therefore, the probability that a peer selects malicious peers as service peers is reduced. Second, we encourage the peer which offers accurate feedbacks by raising its trust value and punish the peer that offers false feedbacks by decreasing its trust value, which is realized by classifying the recommending peers′ ratings toward service provider.

From TBRM[19], we can see that when we take the service provider′s real trust value as the benchmark, there will be a few recommending peers that give too high or too low ratings toward service provider, and most of the recommending peers′ ratings will be similar as the benchmark. Normal distribution can be applied to express this rule, and it is shown in Fig. 2. Supposing there are n recommending peers and $\overline{X}$ is the expectation and σ^2 is the variance of the Normal

distribution, the normal distribution can be established in (10).

$$\overline{X} = \frac{1}{n}\sum_{k=1}^{n} f_{kj}, \quad \sigma^2 = \frac{1}{n}\sum_{k=1}^{n} f_{kj}^2 - \overline{X}^2. \tag{10}$$

According to the distribution function, we classify the recommending peers into two kinds by its trust value, one of which is in interval A: $[\overline{X} - d \times \sigma, \overline{X} + d \times \sigma]$, and the other is outside of interval A, where d is defined in accordance with different environment. Trust value is therefore updated according to (11). For example, if recommending peer k's rating on target peer j belongs to interval A, peer k will be rewarded with an adding value, which is the multiplication of $1-T_{ik}$ with an incentive factor $\mathrm{e}^{-|f_{kj}-\overline{X}|}$. Otherwise, k will be punished with a decreasing value, which is the multiplication of $1 - T_{ik}$ with a punishment factor $1 - \mathrm{e}^{-|f_{kj}-\overline{X}|}$.

$$T_{ik} = \begin{cases} T_{ik} + \mathrm{e}^{-|f_{kj}-\overline{X}|} \times (1 - T_{ik}), & f_{kj} \in A \\ T_{ik} - (1 - \mathrm{e}^{-|f_{kj}-\overline{X}|}) \times (1 - T_{ik}), & f_{kj} \notin A. \end{cases} \tag{11}$$

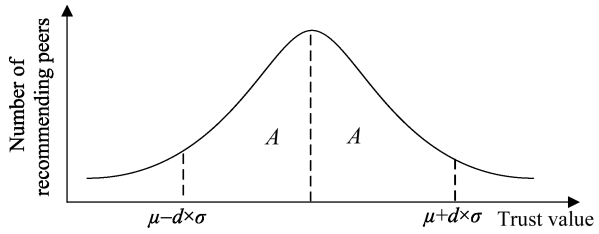


Fig. 2 Distribution of recommending peers

4 Simulation experiments

Several experiments were performed to evaluate the feasibility and effectiveness of METrust in P2P file-sharing system. We choose the same network model used by EigenTrust[4] and construct a Gnutella-like flat unstructured P2P network. The network initially consists of 1 000 nodes interconnected by a power-law distribution with k=2.14. We set 10 000 distinct files in our experimental system. Each peer is assigned to 10 different files randomly and each file is set good or wicked randomly. Each peer is able to issue queries for files, peers can respond to queries, and files can be transferred between two peers to conclude a search process. When a query for a file is issued, it is flooding in the usual Gnutella way, peers that receive the query forward it and check if they are able to respond to it. Then, the list of peers having this file is generated, and the one with the highest reputation is selected to download the desired file. We carried out 100 experiments with 100 downloads in each experiments. Table 1 shows the parameters and default values used in our experiments.

The experiments run on a dual-processor Dell server with Linux kernel 2.6.9. The programs are developed with C++.

Table 1 Parameters and default values used

Parameter	Value
Amount of peers	1000
Count of files	10000
Amount of each peer′s files	10
Direct trust factor δ	0.5
Risk factor α	0.5
d	0.5
Friend peer threshold	0.7
Number of neighbor peers r	6
TTL	6
Time window n	10

The experiments are divided into two groups, one group is to evaluate the performance, and the other is to discuss the security. According to different experiment goals, peers are classified into good peers, general peers, and malicious peers in our experimental system. Good peers provide good files and honest feedbacks. General peers offer good files and give random feedbacks. Malicious peers include simple malicious peers, collusive peers, on-off peers, and oscillating peers. Simple malicious peers provide wicked files and dishonest feedbacks. Collusive peers offer good files and give dishonest feedbacks to exaggerate the peers of the same kind and slander good peers. On-off peers provide good files and honest feedbacks to build trust at the beginning, but afterward, they offer wicked files and false feedbacks. Oscillating peers alter their behavior between building and milking trust.

In the following experiments, Non-trust, EigenTrust, and METrust are compared to evaluate the performance and security of METrust. In the scheme with Non-trust, the P2P networks offer no security countermeasures, and each peer randomly selects a peer as service provider. In the scheme of EigenTrust, each peer downloads files according to EigenTrust. While in the scheme of METrust, files are downloaded according to METrust.

4.1 Performance analysis

Performance testing experiments were carried out by comparing successful transaction rate, total package size, and consuming time. Transaction success rate is the ratio of the amount of good files and the amount of all files downloaded in all time windows of an experiment. In these group experiments, the peers include good peers of 40 %, general peers of 20 % and simple malicious peers of 40 %. We carried out 100 experiments with 100 downloads in each experiment.

1) Successful transaction rate

Successful transaction rate is measured, as shown in Fig. 3, from which we can see that due to the experimental randomness, the curves of three schemes are various in different experiments. Nevertheless, the curve of METrust is a little higher than the curve of EigenTrust, and it always stays on the top. This is due to TON and the incentive and punishment mechanism. In METrust, the peers with higher trust value have higher probability of being selected as service peers by constructing TON. Moreover, good peers are encouraged, and malicious peers are punished. However, because of the lack of punishment for malicious peers in

EigenTrust, the transaction success rate is lower than that of METrust. After about 50 experiments, the curves of METrust and EighenTrust become relatively high and keep stable. However, the curve of Non-trust always stays at the bottom and remains horizontal in each experiment. The reason is that there is no trust model, so transaction peers are randomly selected. This experiment demonstrates that comparing with EigenTrust, METrust improves the feasibility and effectiveness.

2) Comparison of total package size

Total package size is adopted to measure the flow in the transmission of the three schemes. Fig. 4 shows the total package size in three schemes. Basically, the three curves increase linearly as the experiment times increase. The curves of METrust and EigenTrust ascend more quickly than Non-trust because more trust computation between peers is needed as the experimental times increase. We note that the total package size of METrust is lower than that of EigenTrust but higher than that of Non-trust. The reason is that EigenTrust calculates a peer′s global trust value iteratively by gathering information in the whole network after each transaction, while METrust is a local trust model and it gets peers′ trust value by querying friend peers only. As there is no need to gather information for the calculation of trust value, the curve of Non-trust always stays at the bottom. The results show that METrust decreases the overheads than that of EigenTrust.

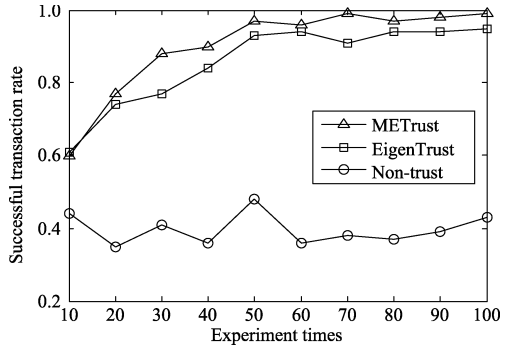


Fig. 3 Comparison of successful transaction rates in three schemes

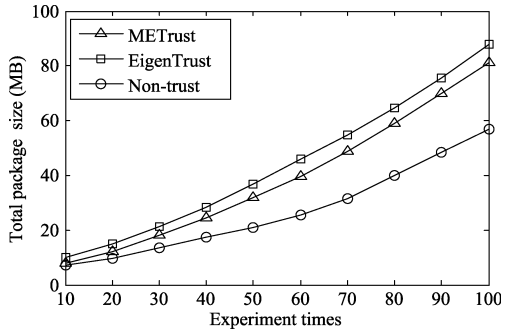


Fig. 4 Comparison of total package sizes

3) Comparison of consuming time

The consuming time is used to measure the time cost in the three schemes. Fig. 5 plots the consuming time as the experiment times increase. Note that the curves of the three schemes nearly reach to linearity with the experiments going on because the same calculation is taken after each download. Non-trust takes less time than the other two owing to the absence of trust model. Moreover, the curve of METrust always stays between that of Non-trust and EigenTrust because METrust does not need to search trust information for target peer in the whole network to compute global trust. This experiment demonstrates that METrust is better than EigenTrust in consuming time.

4.2 Security analysis

The security of METrust is evaluated when there exist free riding and some common attacks, such as simple malicious attack, collusion, on-off attack, and oscillation. There are 100 experiments with 100 downloads carried out in each experiment.

1) Free riding

In this scenario, there were good peers of 40 %, general peers of 20 %, and simple malicious peers of 40 % as mentioned above. A peer was randomly selected as a free rider, which only downloaded files and did not provide any services and feedbacks. The changes of neighbors′ number of good peers and the free rider are shown in Fig. 6. Suppose that the upper limiting number of neighbors of each peer is 6 in our experiment. We can see that when the experiment reaches 30 times, the number of the free rider′s neighbors is 3, while it is 4 for the good peer. When the experiment times reach 50, the number of the free rider′s neighbors is 0, while it is 6 for the good peer. The reason is that the free rider is isolated because of its no contributions. Therefore, METrust can resist free rider effectively.

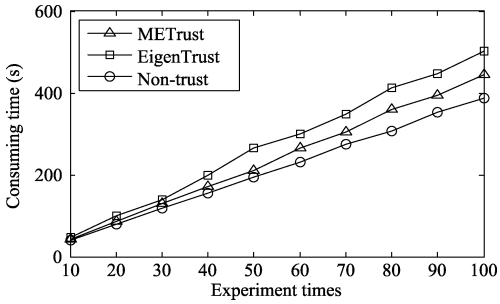


Fig. 5 Comparison of consuming times

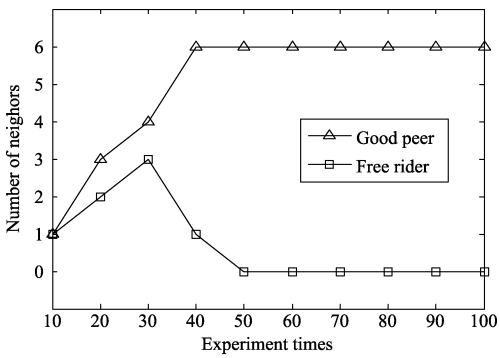


Fig. 6 Comparison of neighbors′ numbers of good peer and free rider

2) Simple malicious attack

In this scenario, we divided peers into good peers and simple malicious peers. The percentage of simple malicious peers varied by increment of 10 %, from 10 % to 90 %. The successful transaction rate after 1000 downloads is observed, as shown in Fig. 7. With the simple malicious peers′ percentage increasing, the curves of METrust, EigenTrust, and Non-trust descend. However, the curve of the METrust is a little higher than EigenTrust and always stays on the top. The curve of Non-trust descends quickly. When the percentage of malicious peers reaches 60 %, downloads from good peers in METrust still keep a higher level and exceed EigenTrust, which is owing to incentive and punishment mechanism. As a result, METrust is a more effective model to restrain malicious behavior than EigenTrust.

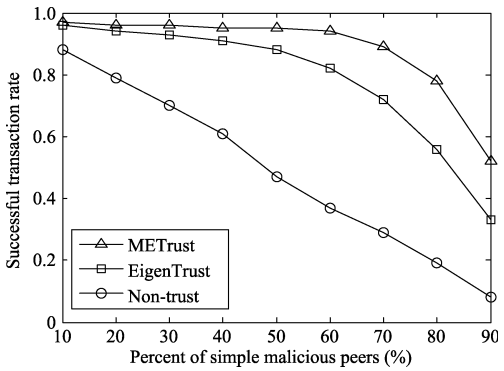


Fig. 7 Comparison of successful transaction rates in different percent of simple malicious peers

3) Collusion

In this scenario, we divided peers into good peers and collusive peers. Collusion league had an evil intention of subverting system by reporting false feedbacks. The ratio of collusive peers varied by increment of 10 %, from 10 % to 50 %. Successful transaction rate is compared among METrust, EigenTrust, and Non-trust to evaluate the efficiency of METrust against collusive attack. In Fig. 8, we can see that the successful transaction rate of METrust is always higher than EigenTrust and declines slowly. The results show that METrust can effectively resist collusive attack. The reason is that the peer′s evaluation of itself for service is introduced in METrust. In this way, it is difficult for the collusive peers to form the collusive league.

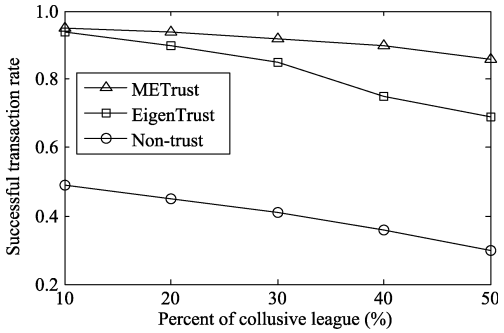


Fig. 8 Comparison successful transaction rates in collusive scenario

4) On-off attack

We selected a peer as on-off peer randomly, which first provided good files and honest feedbacks in the former 50 experiments and then bad files and dishonest feedbacks in the later 50 experiments. The trust values of the peer in 100 experiments are evaluated, as shown in Fig. 9. The trust value is higher than 0.8 and increases gradually in the former 50 experiments. However, with the experiments going on, the curve descends greatly due to the bad files given by the peer. The reason is that when the peer offers good files, the trust value will increase by incentive mechanism. Otherwise, the trust value will decrease by punishment mechanism.

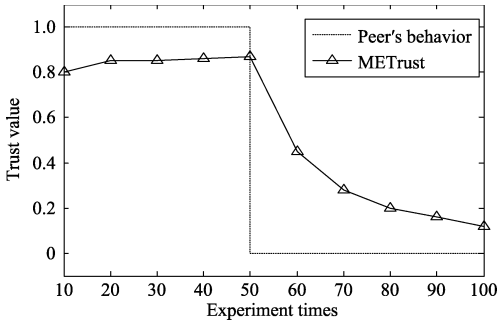


Fig. 9 Change of on-off peer′s trust value

5) Oscillation

We simulated an oscillating peer selected randomly. Fig. 10 shows the computed trust value of the oscillating peer between building and milking trust with the frequency of 20 experiments. First, the peer provides good files and honest feedbacks, so the trust value is high. Then, its trust value decreases greatly when it milks trust by giving bad files. We note that the rebuilding of trust value is slow, that is, the oscillating peer cannot simply increase its trust value quickly by acting well within a short period, which is owing to the incentive and punishment mechanism. As the experiments go on, the more the milking behaviors are, the more slowly the trust value builds back. This experiment shows METrust can detect the oscillating behavior in time and resist strategically changed action of malicious peers.

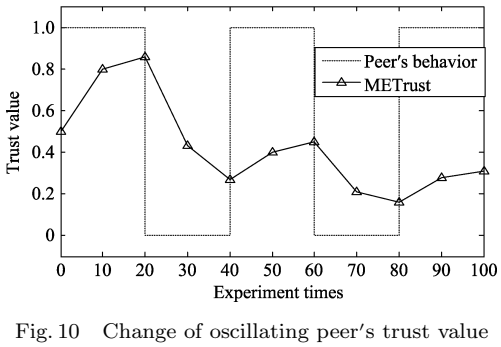


Fig. 10 Change of oscillating peer′s trust value

From experimental results above, we can see that METrust can effectively defend uncooperative and several malicious behaviors. The main tradeoff is the maintaining cost of TON. In addition to the communication cost

of retrieving required information for each trust computation, more messages are sent to adapt neighbor peers and maintain TON. Another tradeoff is the storage cost. In METrust, each peer maintains the mutual evaluation information of the historical transactions and the reputation information of neighbor peers. If the system scale increases and each peer interacts with a large number of peers, the storage cost will increase rapidly. However, this can be controlled by the size of time window, which means we can take the storage cost under control. During trust computation, the major cost is computing the mutual evaluation consistency for transactions in a time window and the similarity risk of recommendations. These depend on the length of the evaluation list. The computational complexity is $O(L)$. Parameter L is the length of evaluation list. Compared with the processing capacity of current computer systems, the computational cost of METrust is negligible.

5 Conclusions and future work

In this paper, a novel trust model called METrust is proposed for P2P systems, in which trust value is stored in distributed way and aggregated by the direct trust and the recommendation trust. First, neighbor relationships are established by TON, which improves the accuracy of trust computing and can resist collusion effectively. Second, the direct trust is computed by mutual ratings and time decay function after a direct transaction. The mutual evaluation factor is adopted to calculate time decay function objectively. Third, in the recommendation trust computation, similarity and similarity risk are used as the credibility measure to aggregate the ratings from recommending peers. At last, incentive and punishment mechanism is adopted to improve the initiative of peers, encourage good behavior and punish malicious peers. The experimental results show the effectiveness of METrust in performance and security against various malicious peers.

Future work will focus on improving the robustness and adaptability of trust model. In the actual network environment, the behaviors of malicious peers are much more complicated and a combination of almost any malicious peers can create a new malicious act[20]. We are investigating different threat models in distributed computing environments and exploring mechanisms to make METrust more robust against malicious behaviors. Also, we aim to extend and implement METrust in other distributed systems, such as Ad Hoc network and wireless sensor networks (WSN), etc. In such resource-limited environments, the trust management model is another concern. How to effectively utilize the limited computing, storage, communication, or energy resources of peers to improve the survivability of the Ad Hoc network or WSN is worthy of further research.

References

[1] M. Karakaya, I. Korpeoglu, O. Ulusoy, B. U. Ankara. Free riding in peer-to-peer networks. *Internet Computing*, vol. 13, no. 2, pp. 92–98, 2009.

[2] H. F. Deng, W. Deng, H. Li, H. J. Yang. Authentication and access control in RFID based logistics-customs clearance service platform. *International Journal of Automation and Computing*, vol. 7, no. 2, pp. 180–189, 2010.

[3] C. W. Chang, H. Pan, H. Y. Jia. A secure short message communication protocol. *International Journal of Automation and Computing*, vol. 5, no. 2, pp. 202–207, 2008.

[4] S. D. Kamvar, M. T. Schlosser, H. Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th international World Wide Web conference*, ACM, Budapest, Hungary, pp. 640–651, 2003.

[5] R. F. Zhou, K. Hwang. PowerTrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 460–473, 2007.

[6] R. F. Zhou, K. Hwang, M. Cai. GossipTrust for fast reputation aggregation in peer-to-peer networks. *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 9, pp. 1282–1295, 2008.

[7] L. Xiong, L. Liu. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, 2004.

[8] M. Srivatsa, L. Xiong, L. Liu. TrustGuard: Countering vulnerabilities in reputation management for decentralized overlay networks. In *Proceedings of the 14th World Wide Web Conference*, ACM, Chiba, Japan, pp. 422–431, 2005.

[9] J. L. Hu, Q. Y. Wu, B. Zhou, J. H. Liu. Robust feedback credibility-based distributed P2P trust model. *Journal of Software*, vol. 20, no. 10, pp. 2885–2898, 2009.

[10] Z. G. Shi, J. W. Liu, Z. L. Wang. Dynamic P2P trust model based on time-window feedback mechanism. *Journal on Communications*, vol. 31, no. 2, pp. 120–129, 2010. (in Chinese)

[11] C. Q. Tian, S.H. Zou, W. D. Wang, S. D. Cheng. A new trust model based on recommendation evidence for P2P networks. *Chinese Journal of Computers*, vol. 31, no. 2, pp. 270–281, 2008. (in Chinese)

[12] L. D. Huang, G. Xue, X. L. He, H. L. Zhuang. A trust model based on evidence theory for P2P systems. *Applied Mechanics and Materials*, vol. 20–23, pp. 99–104, 2010.

[13] J. Wang, H. J. Sun. A new evidential trust model for open communities. *Computer Standards and Interfaces*, vol. 31, no. 5, pp. 994–1001, 2009.

[14] Y. F. Wang, A. Nakao. Poisonedwater: An improved approach for accurate reputation ranking in P2P networks. *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1317–1326, 2010.

[15] C. Q. Tian, J. H. Jiang, Z. G. Hu, F. Li. A novel super-peer based trust model for peer-to-peer networks. *Chinese Journal of Computers*, vol. 33, no. 2, pp. 345–355, 2010. (in Chinese)

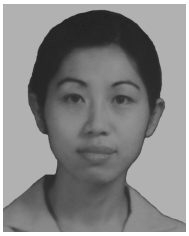
[16] N. Fedotova, L. Veltri. Reputation management algorithms for DHT-based peer-to-peer environment. *Computer Communications*, vol. 32, no. 12, pp. 1400–1409, 2009.

[17] C. Selvaraj, S. Anand. A role based trust model for peer to peer systems. In *Proceedings of the 2009 International Conference on Computer and Network Technology*, Chennai, India, pp. 50–54, 2010.

[18] C. Y. Niu, J. Wang, R. M. Shen. A trust-enhanced topology adaptation protocol for unstructured P2P overlays. In *Proceedings of the 3rd International Conference on Semantics, Knowledge and Grid*, ACM, Xi'an, PRC, pp. 200–205, 2007.

[19] Y. M. Lliu, S. B. Yang, L. T. Guo, W. M. Chen, L. M. Guo. A distributed trust-based reputation model in P2P system. In *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, IEEE, Qingdao, China, vol. 1, pp. 294–299, 2007.

[20] S. S. S. Sivatha, S. Geetha, M. Marikannan, A. Kannan. A neuro-genetic based short-term forecasting framework for network intrusion prediction system. *International Journal of Automation and Computing*, vol. 6, no. 4, pp. 406–414, 2009.



Chun-Ling Cheng received the B. Sc. degree in computer software and M. Sc. degree in computer application from Nanjing University of Science and Technology, Jiangsu, PRC in 1993 and 1996, respectively. She is currently an associate professor at the College of Computer, Nanjing University of Posts and Telecommunications.

Her research interests includes covers network and information security.

E-mail: chengcl@njupt.edu.cn (Corresponding author)

Xiao-Long Xu received the M. Sc. and Ph. D. degrees in computer science and technology from Nanjing University of Posts and Telecommunications, Jiangsu, PRC in 2002 and 2008, respectively. He is currently an associate professor at the College of Computer, Nanjing University of Posts and Telecommunications.

His research interests include computer software, distributed computing, and Information Security.

E-mail: xuxl@njupt.edu.cn

Bing-Zhen Gao received the B. Sc. degree in mathematics from Shandong Normal University, PRC in 2007. She is currently working toward the M. Sc. degree in computer application at Nanjing University of Posts and Telecommunications.

Her research interests include information security and P2P network.

E-mail: qianrui25@126.com