SMART: A Subspace based Malicious Peers Detection algorithm for P2P Systems

Xianglin Wei[1, 2], Jianhua Fan[1], Ming Chen[2], Tarem Ahmed[3], and Al-Sakib Khan Pathan[3]

[1]Nanjing Telecommunication Technology Institute, Nanjing, China
[2]Department of Computer Science and Engineering, PLA University of Science and Technology, Nanjing, China
[3]Department of Computer Science, International Islamic University Malaysia, Kuala Lumpur, Malaysia
wei_xianglin@ieee.org, cm@plaust.edu.cn, sakib@iium.edu.my

***Abstract*:** In recent years, reputation management schemes have been proposed as promising solutions to alleviate the blindness during peer selection in distributed P2P environment where malicious peers coexist with honest ones. They indeed provide incentives for peers to contribute more resources to the system and thus promote the whole system performance. But few of them have been implemented practically since they still suffer from various security threats, such as collusion, Sybil attack and so on. Therefore, how to detect malicious peers plays a critical role in the successful work of these mechanisms, and it will also be our focus in this paper. Firstly, we define malicious peers and show their influence on the system performance. Secondly, based on Multiscale Principal Component Analysis (MSPCA) and control chart, a Subspace based MAlicious peeRs deTecting algorithm (SMART) is brought forward. SMART first reconstructs the original reputation matrix based on subspace method, and then finds malicious peers out based on Shewhart control chart. Finally, simulation results indicate that SMART can detect malicious peers efficiently and accurately.

***Keywords*:** P2P, Multiscale Principal Component Analysis, Shewhart control chart, malicious peer.

1. Introduction

In order to stimulate peers to contribute resources and assist peers to select the most trustworthy collaborators, several reputation management schemes have been proposed [1], [2]. These schemes try to evaluate the transactions performed by peers and assign reputation values to them to reflect their past behavior features. And these reputation values will be the basis for identifying trustworthy peers to reduce the blindness of peer selection. Although these schemes have been proved to be theoretically attractive, they still have a long way to go before practical deployment. Because they are still faced with various attacks including self-promoting, whitewashing, slandering, collusion [3] and Sybil attack [4]. To simplify the description, these P2P systems with reputation management schemes will be referred to as Reputation based P2P (RP2P) systems for short, and those peers who initiate attacks will be referred to as malicious peers, other peers beside malicious ones will be called honest peers.

As a burgeoning field, malicious peer detection has attracted the attention of many researchers in the recent years. A detector algorithm is proposed in [5] to find liar peers that send wrong feedback to subvert reputation system. Ji et al. suggested a group based metric for protecting P2P network against Sybil Attack and Collusion by dividing the whole network into some trust groups based on global structure information which is hard to obtain [6]. Recently, an upload entropy scheme is developed by Liu et al. to prevent collusions and further enhance robustness of private trackers sites [1]. But the threshold of this scheme needs to be selected by experiment. Moreover, Lee et al. put forward a simplified clique detection method to detect the colluders [7], but their method is restricted to colluders forming a clique.

Many of these methods either concentrated on malicious peers of some particular categories or are based on global assumption, in this work, however, we focus on developing a general Subspace based MAlicious peeRs deTecting algorithm (SMART). The main differences between SMART and existing methods are: on the one hand, SMART aims at detecting malicious peers of multi-categories rather than some particular categories; on the other hand, SMART is based only on reputation information rather than global structure information.

The rest of the paper is organized as follows. Related work is summarized in Section 2, Section 3 illustrates the influence of malicious peers and introduces SMART, and in Section 4 many experiments are conducted to evaluate the performance of SMART. Finally, we conclude our main works and mention further research directions in Section 5.

2. Related Work

Mekouar et al. proposed a Malicious Detector Algorithm in [5] to detect liar peers that send wrong feedback to subvert reputation system. That is, after each transaction between a pair of peers, both peers are required to generate feedback to describe the transaction. If there is an obvious gap between the two pieces of feedback, both are regarded being suspicious. Ji et al. raised a group based metric for protecting P2P network against Sybil attack and collusion by dividing the whole network into some trust groups based on global structure information which is hard to obtain [6]. In [3], Lian et al. recommended various collusion detection approaches including pair-wise detector and traffic concentration detector with data of Maze file sharing application based on trace analysis. In order to guarantee the correctness of the reputation calculation, Despotovic et al [8] compared the probabilistic estimation and social network methods. Besides, they also identified four classes of collusive behavior. Recently, Tehale et al used the false message concept for identifying and verifying the Sybil nodes in the network [28]. Selvaraj et al presented a comprehensive survey of security issues in Reputation Management Systems for P2P networks in [29]. Jin et al proposed a peer based monitoring method in Peer-to-Peer Streaming environment [30]. Koutrouli et al provided a thorough view of the various credibility threats

against a decentralized reputation system and the respective defense mechanisms [31].

Recently, an upload entropy scheme is developed by Liu et al. to prevent collusions and further enhance robustness of private trackers' sites [1]. But the threshold of this scheme needs to be settled manually. Moreover, Lee et al. put forward a simplified clique detection method to detect the colluders [7], but their method is restricted to colluders who form a clique. Ciccarelli et al [9] surveyed the literature on P2P systems security with specific attention to collusion, to find out how they resist to such attacks and what solutions can be used. On the one hand, they summarized five collusive categories, and then investigated the influence of collusion on various applications. On the other hand, they discussed the feasible solutions that can be utilized to resist collusions, such as game theory and so on. Liu et al [10] brought forward a new strategy based on trust value and considers both the quality and the number of shared resources to avoid the phenomenon of free riding. Moreover, they also sketched collusion, slander and other misbehavior during strategy design. A MSPCA and Quality of Reconstruction based method PeerMate was proposed in our former work [11], it can efficiently detect malicious peers for P2P systems. However, PeerMate cannot find out malicious peers which initial Sybil attack to the system. Moreover, PeerMate needs a reconstruction threshold, which can remarkably impact its efficiency.

Besides, many micropayment systems based methods have been proposed to help the P2P systems resist collusive behavior, in this paper, however, we mainly focus on how to detect malicious peers under P2P systems with reputation management schemes.

3. Subspace based Malicious Peers Detection

Firstly, we present the detecting context GRep, which is derived from current P2P systems. Secondly, malicious peers are divided into several categories and then their influence on the system performance is illustrated. Finally, SMART is introduced.

3.1 Detecting context

Before designing the detection algorithm, we first describe the detection context, which is derived from current RP2P systems, such as TVTorrents (www.tvtorrents.com) **Error! Reference source not found.**, EigenTrust [2] and Maze (http://maze.tianwang.com) [13]. In this context, the content exchange process obeys the typical P2P workload models and is divided into several time slots (rounds). During each round, each peer initiates requests and the request process follows some typical P2P workload model, such as the workload model in KaZaA [14] and the BitTorrent workload model [1]. Moreover, we have found that the effectiveness of SMART is independent of the underlying workload model used. For the sake of simplicity, we adopt the typical model in literature [14], which is detailed in Section 4. Moreover, each peer is assigned an initial reputation value, which will increase by X_u when it uploads a piece of valid content and decrease by X_d when downloading a valid piece, and $X_u \geq X_d$.

Reputation matrix. Let N be the total number of peers and X_p^T be the reputation value of peer p at the end of the T^{th} round, $1 \leq p \leq N$. Consequently, the reputation value of all the peers can form a reputation vector $XV^T = (X_1^T, X_2^T, \ldots, X_N^T)$ at the end of the T^{th} round. Besides, from the perspective of one single peer p, X_p^t, $1 \leq t \leq T$, can form a reputation time series $XS_p = (X_p^1, X_p^2, \ldots, X_p^T)$. Then we can obtain a reputation matrix $X^{T\times N}$ as in (1) at the end of the T^{th} round. The i^{th} column of $X^{T\times N}$ is the reputation time series of peer i. And the t^{th} row of $X^{T\times N}$ is the reputation vector at the end of round t.

$$X^{T\times N} = \begin{bmatrix} X_1^1 & X_2^1 & \cdots & X_N^1 \\ X_1^2 & X_2^2 & \cdots & X_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ X_1^T & X_2^T & \cdots & X_N^T \end{bmatrix} \quad (1)$$

Reputation matrix retrieval. In centralized RP2P systems, $X^{T\times N}$ is usually collected and stored by a centralized facility, such as the tracker servers or the central server in Maze. In contrast, $X^{T\times N}$ can be collected and calculated by each peer respectively in RP2P systems with decentralized reputation management scheme [2][15]. Hence, at least in some way, we can always get $X^{T\times N}$. For simplicity, we will use $\breve{X}$ to represent $X^{T\times N}$ in the following analysis.

3.2 Malicious peers and their influence on the system

3.2.1 Malicious peers

According to their different behavior features, the malicious peers can be divided into various categories, and hence it is hard to summarize all the categories comprehensively due to the complexity of the behaviors. Here, we mainly focus on the following categories and evaluate our algorithm based on these categories.

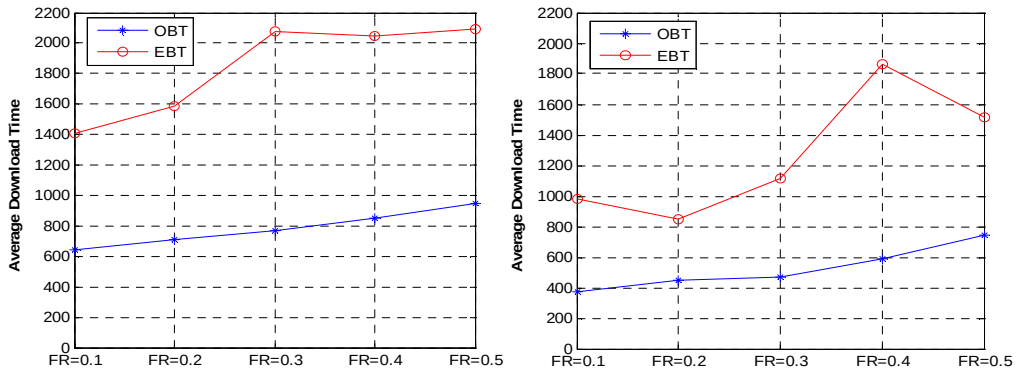
MP1: peers that utilize P2P's resources without providing appropriate amount of resources (i.e., free-riders), such as BitTyrant and BitThief clients, this is because many peers are only in pursuit of maximizing their own profit while lack enthusiasm for contributing services to the entire system.

MP2: peers that upload inauthentic objects to persecute the community, such as the peers controlled by the music industry which inject fake files to KaZaA, this is mainly due to the fact that many contents shared in the P2P community are copyrighted materials, such as latest movies or software, which violates the copyright owners' profit.

MP3: peers that collude with each other, they can be organized to a collusive group or chain through collaborating with each other to promote their reputation values or to decrease other peers' reputation values, such as the colluders in Maze or eBay system;

MP4: peers that create Sybil peers [16] to promote their own reputation values, and hence they can consume more resources in the system, such as the peers in eBay system with fake feedbacks from their Sybil peers.

MP5: peers that exploit P2P's resources for their malicious purposes like worm dispatching, denial of service and so on [17].



(a) The results under FCS (b) The results under EDS

Figure 1. The influence of FRs on the average download time of OBT and EBT

We recognize that this partition is incomplete and there also exist some other malicious peers with more complex behaviors. For example, some peers may belong to multi categories at the same time and their behavior is a combination of the behaviors of multi categories. Besides, a peer is being nice until its reputation is high, and from then on exploits the system. Moreover, the collusive model targeting at some particular reputation management systems maybe more complex than peers belong to MP3 discussed above, such as the collusive models discussed in literature [9] and [8]. These malicious peers are called strategic malicious peers, which will be discussed further in Section 4. Although we mainly focus on the categories from MP1 to MP5, these categories can be used to evaluate our method and the evaluation results not only provide fundamental insight to the malicious peer detecting problem, but also can serve as a benchmark to evaluate the forthcoming detect algorithms, and inspire new detecting algorithms in the future. Furthermore, we assume honest peers count for the majority of all the peers.

3.2.2 The Influence of Malicious Peers on the System

In order to make a straightforward understanding of the influence of the malicious peers, we have conducted some experiments. Concretely, we investigate the influence of peers which belong to **MP1** (i.e. free-riders) on both original BitTorrent (OBT) and BitTorrent system with a typical incentive scheme named EigenTrust [2] (EBT). The experiments are conducted on the simulator developed by Legout et al [18][19], moreover, we implement EigenTrust on it, replace its homogeneous assumption of peers' upload capacities with heterogeneous one [15] and add an information collection module to it. To make the result more general , we investigate the influence of free-riders under two scenarios: in the first scenario, peers arrive in flash crowd fashion (i.e. all peers arrive simultaneously), while the rate at which peers join the torrent decreases exponentially with time under the second scenario. For the sake of simplicity, the flash crowd scenario and the exponential decreasing scenario will be referred to as FCS and EDS respectively in the following analysis. Other settings can be referred to [15][19]. In this simulation, the download time of a peer is defined as its download completion time minus its arrival time, and the Average Download Time (ADT) is defined as the average value of all peers' download times. We choose ADT as our metric to investigate the influence of malicious peers. We vary the FRs from 0.1 to 0.5 of the system while leave other parameters intact, and show the results in Fig. 1, which are averaged over 20 runs. Fig. 1(a) and (b) show the result under FCS and EDS respectively, and we can make two observations from them: Firstly, the download times of the peers tend to increase as the FR increases from 10% to 50% under both OBT and EBT system; Secondly, EBT increases the download time of OBT although it bring some trust into the system.

In summary, under both scenarios, the more the free-riders the system has, the higher the ADT will be. In fact, as the simplest form of the malicious peers, the free-riders might have already hurt the system performance, let alone those peers with more complex malicious behaviors. Consequently, we need to find out or even punish the malicious peers in order to promote the system performance.

3.3 Problem statement and fundamental idea of SMART

As illustrated before, all the malicious peers are with various objectives when joining the system. Despite of this, they possess an identical feature, which also differentiates them from honest ones, i.e. they behave differently from honest peers. Since the reputation value of a peer reflects its behavior features, different behaviors will lead to different reputation values, which will afterward lead to their different reputation time-series in $\tilde{X}$. Therefore, we can distinguish malicious peers from honest ones if we can extract their different behavior features, which are embedded in the different deterministic features of their reputation time-series in $\tilde{X}$.

Based on this observation and inspired by the algorithms on anomalies detecting [20][21][22], we bring forward SMART based on subspace separation and control chart. More concretely, SMART first separates the original *T*-dimensional space into honest subspace and malicious subspace based on Multiscale Principal Component Analysis (MSPCA), and then reconstructs the reputation matrix based on the honest subspace, finally applies Shewhart control chart [23] on the reconstruction error matrix to find out the malicious peers.

3.4 SMART

3.4.1 MSPCA based Reputation Matrix Reconstruction

MSPCA. MSPCA combines the ability of PCA to de-correlate the variables by extracting a linear relationship, with that of wavelet analysis to extract deterministic features

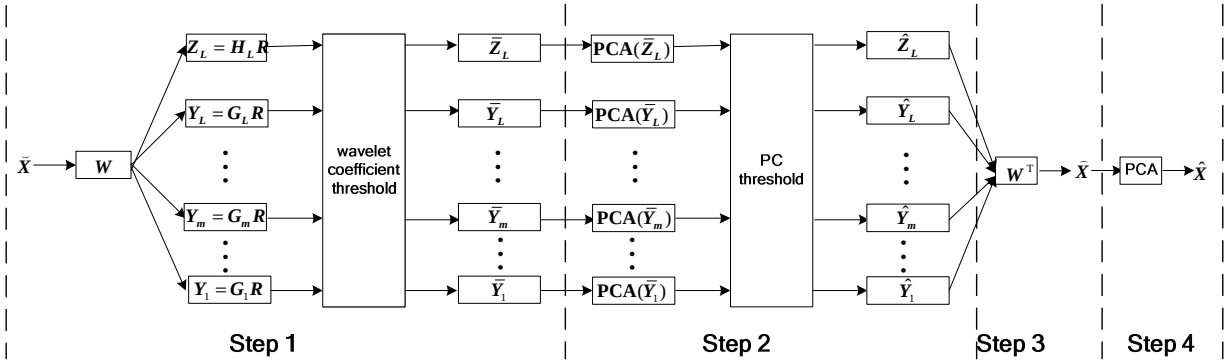


Figure 2. Four steps of MSPCA

and approximately de-correlate auto-correlated measurements [24]. Consequently, MSPCA is used to reconstruct the reputation matrix. After adding wavelet coefficients denoising process to the MSPCA proposed in [24], the MSPCA used contains four steps. For the sake of clarity, this process is illustrated in Fig. 2.

Step 1: Wavelet decomposition of $\breve{X}$: apply wavelet decomposition $\boldsymbol{W}$ to each column of $\breve{X}$ to get wavelet coefficient matrix Z_L, Y_m (m=1, …, L) at each scale; then filter the wavelet coefficients according to MAD method [25] and arrive at $\bar{Z}_L$, $\bar{Y}_m$ (m=1,…,L);

Step 2: Principal component analysis of wavelet coefficient matrix: firstly, apply PCA to wavelet coefficient matrix $\bar{Z}_L$, $\bar{Y}_m$ (m=1,…, L) at each scale; secondly, select the number of principal components reserved according to scree plot method [26]; finally, reconstruct the wavelet coefficients matrix $\hat{Z}_L$, $\hat{Y}_m$;

Step 3: Wavelet reconstruction of the reputation matrix: reconstruct the matrix based on $\hat{Z}_L$, $\hat{Y}_m$ (m=1,…,L) through inverse wavelet transformation $\boldsymbol{W}^{\mathrm{T}}$ and obtain $\hat{X}$;

Step 4: Principal component analysis of reconstructed matrix: apply PCA to $\hat{X}$ to reduce the dimensionality and then obtain reconstructed matrix $\hat{X}$.

Wavelet denoising. In MSPCA, wavelet denosing is applied to $\breve{X}$ in Step 1 to eliminate the influence of inaccurate or noise data in $\breve{X}$. Moreover, during the second step, the soft threshold function is used as defined by Donoho in [26].

PCA based matrix reconstruction. As a typical multivariate statistical analysis technique, PCA (Principal Components Analysis) is a general method to find pattern in high-dimensional data and has been widely used in many fields, such as pattern recognition and data compression. Moreover, PCA has also been used to detect traffic anomalies by separating the principal components to normal and anomalies subspace, and then the anomalies part is used to detect traffic anomalies [20]. Inspired by this detection method, PCA is used to transform the matrix $\hat{X}$ into two subspaces: the honest subspace and the malicious subspace. More concretely, after applying PCA to $\hat{X}$, the first r principal components are selected to construct the honest subspace since they captures most of the variances of $\hat{X}$, while the last N-r principal components are used to construct the anomalies subspace. And then $\hat{X}$ is obtained after projecting $\hat{X}$ onto the honest subspace. Similarly, $\tilde{X}$ is obtained through projecting $\hat{X}$ onto the malicious subspace. Moreover, $\hat{X}$ and $\tilde{X}$ satisfy the equation $\hat{X} = \tilde{X} + \hat{X}$. To make it simple, each column i of $\tilde{X}$ is called the reconstruction error time series of peer i.

In general, the reputation values of honest peers are mainly enclosed in the honest subspace since their time-varying patterns are closer to the first r principal components than those of malicious peers, while the reputation values of malicious peers are expressed more by the malicious subspace. Consequently, after reconstruction, the changes of the reputation time-series of malicious peers are larger than those of honest peers, in other words, the reputation values of malicious peers in $\tilde{X}$ are larger than that of honest ones. And this can help us distinguish malicious peers from honest ones.

3.4.2 Shewhart Control Chart based Malicious Peers Detection

Here we treat $\tilde{X}$ as a sample of a production process, and each column (i.e. a reconstruction error time-series of a peer) of $\tilde{X}$ as a sample subgroup, there will be a substantial change between two subgroups if one of them is a reconstruction error time-series of an honest peer and the other one is a time-series of a malicious peer, since the reconstruction error of malicious peers in $\tilde{X}$ are larger than that of honest ones. Consequently, we adopt Shewhart R control chart [23] to find out malicious peers, which is good at detecting the change between sample subgroups.

Let the mean of the production process be μ and the standard deviation be σ. Then the central line (CL), the upper control limit (UCL) and the lower control limit (LCL) are fixed at:

$$\mathrm{UCL} = \mu + k\sigma \tag{2}$$

$$\mathrm{CL} = \mu \tag{3}$$

$$\mathrm{LCL} = \mu - k\sigma \tag{4}$$

where k is the distance of the control limits from the central line, expressed in standard deviation units. According to central limit theorem, k is usually chosen as 3. Generally speaking, μ and σ are unknown and are needed to be estimated through samples $\tilde{X}$.

In this work, we have N sample subgroups, $\tilde{X}_i$, i=1, 2, … , N. And there are T samples in each subgroup i. Let the range of each sample subgroup i be R_i, and the estimated control limit can be rewritten as:

$$\mathrm{UCL} = (1 + k\frac{d_3}{d_2})\bar{R} \tag{5}$$

$$\text{CL}= \bar{R} = \frac{1}{N}\sum_{i=1}^{N} R_i \quad (6)$$

$$\text{LCL}= (1-k\frac{d_3}{d_2})\bar{R} \quad (7)$$

where $d_3\bar{R}/d_2$ is the estimator of σ, and $\bar{R}$ is the estimator of μ. After obtaining UCL and LCL, a sample subgroup i (i.e. peer i) is identified as malicious if its R_i is larger than UCL or lower than LCL. Moreover, the values of d_2 and d_3 only depend on T [23].

3.4.3 SMART Algorithm

The pseudo code of SMART is illustrated in **Algorithm** 1. Firstly, in line 1, SMART applies MSPCA to $\breve{X}$ and obtain $\hat{X}$ and $\tilde{X}$. Secondly, in line 2, SMART calculates UCL and LCL according to (5) to (7). Finally, from line 3 to 7, SMART identifies malicious peers according to the control limit.

Algorithm 1 SMART

Input: $\breve{X}$ △ the reputation matrix

Output: **SMPS** △ Suspicious Malicious Peers Set

1: obtain $\hat{X}$ and $\tilde{X}$ after applying MSPCA to $\breve{X}$
2: calculating UCL and LCL according to (5) and (7)
3: **for** i =1 to N
4: **if** R_i>UCL **or** R_i <LCL △ R_i is the range of column i of $\tilde{X}$
5: i is considered as a malicious peer, add i to **SMPS**
6: **end if**
7: **end for**

Time complexity. The time complexity of SMART mainly lies on MSPCA, whose time complexity is $O(TN^2L)$. Therefore, the complexity of SMART is $O(TN^2L)$. Moreover, the storage cost of SMART is $O(TN)$.

4. Simulations and Results

To explore aspects of SMART and compare it with existing algorithms are difficult to study using traces of real systems or analysis, consequently, we use a simulation-based approach for understanding and evaluating SMART and existing algorithms. Such an approach provides the flexibility of carefully controlling the configuration parameters of the various detecting algorithms. This would be difficult or even impossible to achieve using live Internet measurement techniques. Thus, while certain interactions specific to a real deployment will be missed, we believe the abstraction is rich enough to expose most details that are relevant to our experiments. Concretely speaking, after introducing our experimental context; we present the simulation results, and then give a discussion on the results as well as possible usability of SMART.

4.1 Simulation Context and Comparison Method

Here, we adopt the workload model in [14] as the underlying workload model of our simulations. Concretely, the workload is as follows. The contents arrive at constant rate $\lambda_O > 0$ and the popularity of them follows *Zipf* distribution. When a piece of content arrives, its popularity rank is determined by selecting randomly from the *Zipf*(1) distribution. On average, a client requests a constant number of pieces of content per round, choosing which piece of content to fetch from a *Zipf* probability distribution with parameter 1.0. To simplify our model, we assume that all of the content in the system is of equal size. Table 1 describes the parameters setup in the simulation experiment. And the malicious peers are selected randomly from all the peers.

Note that our simulation is from the measurement results from KaZaA rather than BitTorrent workload model [1], since after investigating the simulation results of this workload model, we get similar upload entropy of the system as those in BitTorrent workload model [1]. Consequently, this workload model is sufficient to illustrate the performance of our detecting algorithm.

Table 1. Simulation parameters

Symbol	Meaning	Base value
N	# of peers	200
O	# of contents	4000
λ_R	per-user request rate	2 contents /round
λ_O	content arrival rate	varies
P_M	the ratio of # of malicious peers to # of peers	varies
P_h	the honest possibility that strategic malicious peer act as honest ones	varies

4.2 Comparison Benchmarks and Evaluation Metrics

Comparison benchmarks. We choose three existing schemes as comparison benchmarks: EigenTrust, Upload Entropy (UEntropy) schemes [1] and our former algorithm PeerMate [11]. In EigenTrust, iterative calculation is implemented to obtain each peer's global reputation value and peers with the lowest reputation values are treated as the least trustworthy peers, which therefore will be treated as malicious peers distinguished by EigenTrust scheme in our simulation. The second scheme aims at stimulating peers to share content in Private BT society. And those peers with lowest upload entropy will be considered as the least trustworthy collaborators, in other words, they are the suspicious malicious peers. Therefore, in order to guarantee the fairness of comparison, in UEntropy scheme, those peers with the lowest entropy will be treated as suspicious malicious peers found by UEntropy. PeerMate detects malicious peers based on MSPCA and Quality of Reconstruction (QR).

Evaluation metrics. Let **MPS** (Malicious Peers Set) be the malicious peers set, **HPS** (Honest Peers Set) be the set of honest peers, and **SMP** (Suspected Malicious Peers set) be the malicious peers set found by particular scheme. Then we define two metrics TPR (True Positive Ratio) and FNR (False Negative Ratio) as follows:

TPR=|**SMP**∩ **MPS**|/|**MPS**|;

FNR=|**SMP**∩ **HPS**|/| **HPS**|.

where | | represents the rank of a set, and ∩ stands for the intersection of two sets. Consequently, both TPR and FNR range from 0 to 1.

Simulation scenarios. We consider two typical simulation scenarios here. One is simple and the other is more complex. Under the simple scenario, there are no strategic malicious peers in the system. In contrast, there exist some strategic malicious peers in the system under complex scenario.

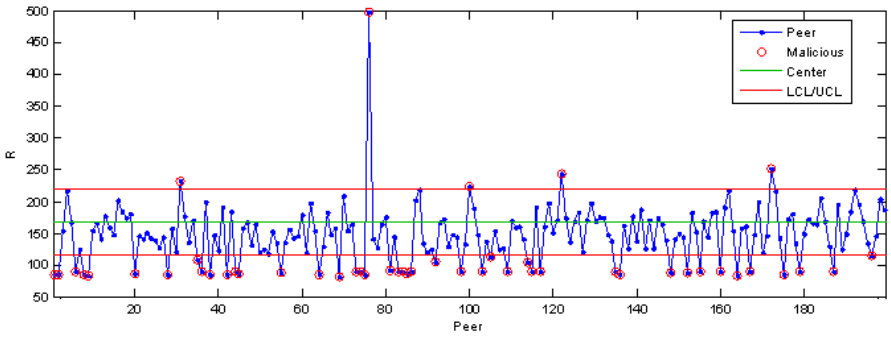


Figure 3. Detecting result of SMART

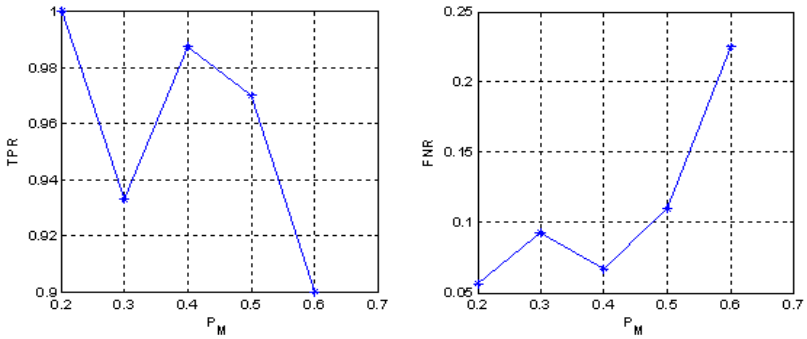


Figure 4. Detecting result of SMART with different P_M

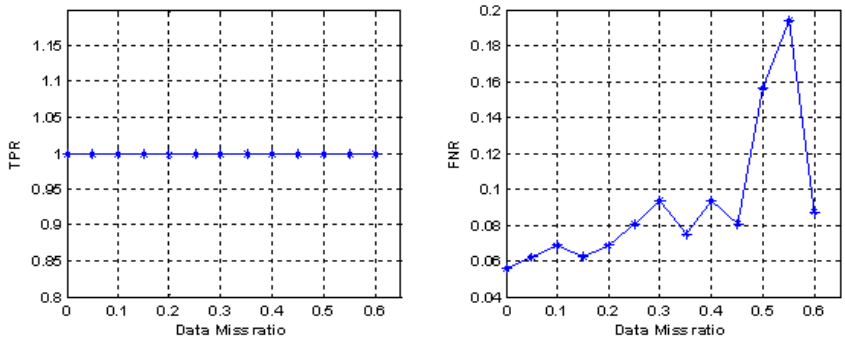


Figure 5. Detecting result of SMART with different data missing ratio

4.3 Simulation Results

4.3.1 Comparison Results of different schemes

Firstly, we compare malicious peers detection results of SMART, EigenTrust, UEntropy and PeerMate with λ_O=2, P_h=0 and P_M=0.2. Other parameters, such as N, O and λ_R are listed in Table 1. After 200 rounds, we can obtain a reputation matrix $\boldsymbol{X}^{200\times200}$, and then we apply the three schemes to $\boldsymbol{X}^{200\times200}$ respectively, the detecting results of SMART is shown in Fig. 3, in which the peers with red circles are malicious peers detected by SMART. Moreover, the detecting results of all the four schemes are shown in Table 2. As noted from Table 2, the TPR of SMART is 100% which is the highest among the four, while the TPRs of EigenTrust, UEntropy and PeerMate are 90%, 57.5% and 97.5% respectively. Besides, the FNR of SMART is 5.63% which ranks the second among the three, while the FNRs of UEntropy, EigenTrust and PeerMate are 10.63%, 2.5% and 7% respectively. Consequently, SMART detects all of the malicious peers with acceptable FNR. The FNR of EigenTrust is 2.5% which is lower than SMART since we only choose the last $N\times P_M$ peers as the malicious peers and this choice helps decrease the FNR of EigenTrust. Moreover, we notice that SMART finds the malicious peers belong to MP4, and this means we can extend SMART in the future to find out Sybil peers in other P2P systems.

Table 2 The detecting results of the four schemes

Schemes	TPR	FNR
EigenTrust	90%	2.5%
UEntropy	57.5%	10.63%
PeerMate	97.5%	7%
SMART	100%	5.63%

4.3.2 Detecting Results of SMART with Different Parameters

Detection results with different P_M. We also investigate the influence of P_M when λ_O =2 and P_h=0. And the results are shown in Fig. 4. From Fig. 4, we can see that the TPR of SMART decreases from 100% to 90% as P_M increases from 20% to 60%, in contrast, the FNR of SMART increases from 5.63% to 22.5%. We also notice that the FNR of SMART is about 10% when half of the peers are malicious. This means the accuracy of SMART's detecting results is acceptable when up to 50% of the peers are malicious.

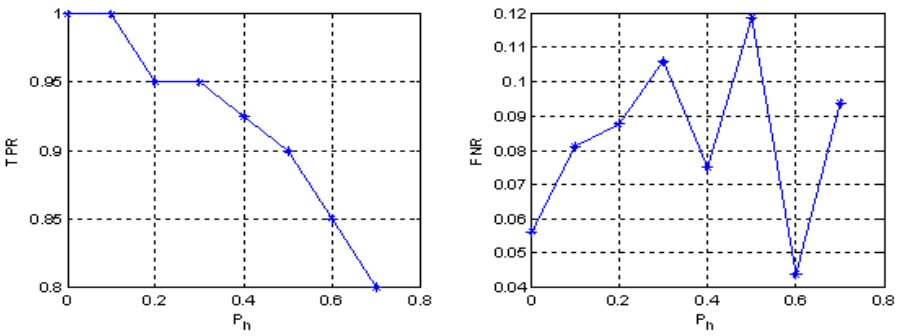


Figure 6. Detecting result of SMART with different P_h

Detection results with missing data. As mentioned before, the reputation values in $\boldsymbol{X}$ may be inaccurate or missing. Consequently, we investigate how SMART could adapt to missing data context with different ratios of missing data from 0 to 60%, while the missing data are selected randomly from $\boldsymbol{X}$. Other parameters are: λ_O =2, P_h=0 and P_M =0.2. For the sake of simplicity, we fix $\boldsymbol{X}$ as follows: if X_i^t is missing, then we set $X_i^t = (X_i^{t-1} + X_i^{t+1})/2$, if 1<t<$T$; $X_i^t = X_i^{t+1}$, if t=1; $X_i^t = X_i^{t-1}$, if t=T. After the 200th round, the results are shown in Fig. 5.

From Fig. 5, we can see that the TPR of SMART keeps at 100% even up to 60% of the elements are missing since data missing cannot change the deterministic features of honest and malicious peers. In contrast, the FNR of SMART increases slowly from 5.63% to 8% as the data missing ratio increases from 0 to 40%, and then increases sharply from 8% to about 21% as the data missing ratio increases from 40% to 55%, at last, the FNR decreases to 11% when the data missing ratio is 60%. This means the performance of SMART is acceptable when the data miss ratio is lower than 40%.

4.3.3 Detection Results of SMART under Complex Scenario

Possibility model. In order to avoid being detected, during each round, many strategic malicious peers will act as honest ones with certain possibility of P_h. Therefore, we investigate SMART with λ_O =2, P_M =0.2 and P_h =0.1, 0.2, 0.3, 0.4, 0.5, 0.6 and 0.7 respectively. The results are shown in Fig. 6.

Fig. 6 demonstrates that the TPR decreases from 100% to 80% as P_h increases from 0 to 70%, while the FNR of SMART fluctuates between 4.5% and 12%. In order to obtain high accuracy and low false alert, the performance of SMART is acceptable when P_h is lower than 40%.

Mixture model. We also evaluate SMART with strategic malicious peers which belong to multi categories at the same time and whose behaviors are a combination of the behaviors of multi categories discussed above. Generally speaking, the mixture of malicious behavior cannot change the essential difference between the behaviors of malicious peers and honest ones. Concretely, we add a few malicious peers whose behaviors are as follows. They act as the behaviors of MP1, MP2, MP3, MP4 and MP5 with certain possibility. After 200 rounds, we find that the TPR of SMART is 95% with FNR equals to 8.3%. This means SMART is also good at finding out malicious peers with mixture behaviors since mixture behaviors cannot change the deterministic features of honest and malicious peers. Here, we leave malicious peers with more complex behaviors for future work since it is hard, if not impossible, to enumerate all of them.

4.4 The Influence of SMART on the System Performance

If we can find out malicious peers with SMART, peers can choose more reliable service providers during peer selection process. Therefore, we compare the request success rate of the system with three different peer selection policies. With the first policy, peers select service providers randomly, while the peers select the provider with the highest reputation value calculated by EigenTrust in the second policy, and the honest peers found out by SMART are selected as the service providers in the third policy. Without loss of generality, the request success rate of each round is defined as the number of successful object transactions divided by the total number of object transactions during this round. And the results with λ_O =2, P_M =0.2 are shown in Fig. 7.

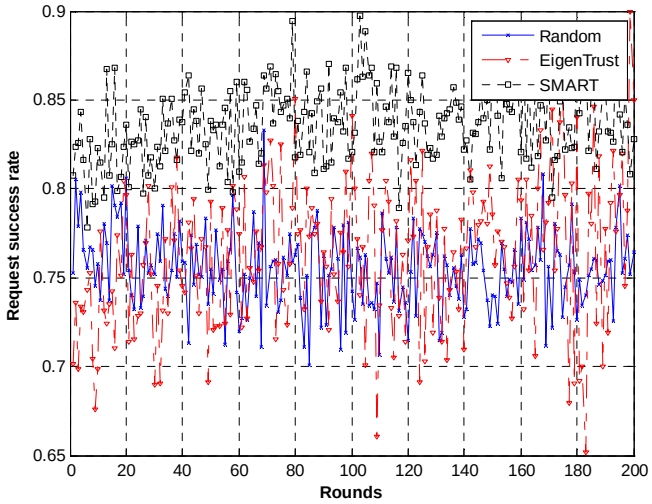


Figure 7. The request success rate of the system with different peer selection policies

From Fig. 7, we can see that SMART achieves the highest request success rate among the three, and its average request success rate of all the rounds is 83.64%. EigenTrust ranks the second, and its average request success rate is 76.31%, while the average request success rate with random peer selection policy is 75.50% and is the worst among the three peer selection policies. And this result shows the potential that SMART can be integrated with existing peer selection algorithms to promote the quality of service or performance of the system. For example, during the choking process in BitTorrent, the malicious peers discriminated by SMART should not be unchoked.

4.5 Discussion

Summary. From these simulations, we can draw the following conclusions: first, aiming at these malicious peers defined in Section 3, SMART can achieve high detecting accuracy even under strategic and data missing context; second, SMART can distinguish malicious peers of **MP4** from honest ones, this may provide an useful inspiration of new algorithms to defend Sybil attack; third, each reputation management scheme needs to implement some algorithms to detect malicious peers to ensure its success. Finally, the detection performance of SMART is not restricted to malicious peers of some particular categories or some particular workload models, as long as the malicious peers have different behaviors from honest peers.

Possible worst case. As the basic idea of SMART is that malicious peers have different behaviors from honest peers. Consequently, some of the malicious peers can guess the reputation change mode of honest peers and make their reputation time series has similar change mode through choosing their behavior of each round carefully. This will decrease the effectiveness of SMART, but if every object transaction is tractable, the malicious peers should contribute enough resources to the system to get the desirable reputation time series, and this will increase the attack cost of these malicious peers.

SMART applications. SMART can be applied to Maze-like and EigenTrust-like systems directly since they have common context. Besides, SMART can also be applied to other RP2P systems if they implement some schemes to collect or compute the reputation values of all the peers, such as iRep [15].

5. Conclusions and Future Work

In this paper, we present SMART, a novel malicious peer detection algorithm for RP2P systems, which distinguishes malicious peers from honest ones by combining MSPCA with Shewhart control chart. Simulation results indicate that SMART achieves high detection accuracy and flexibility on the malicious peers defined in this paper. As a future task, we are planning to extend SMART to make it adaptable to real-time online detection in RP2P systems. Besides, to this end, we omit some malicious peers with more complex behaviors, such as RepTrap attack [27], which are resource consuming for the attackers. We will take them into consideration in our future works.

Acknowledgement

This research was supported in part by the National Natural Science Foundation of China under Grant No. 61070173, Jiangsu Province Natural Science Foundation of China under Grant No. BK2010133 and Jiangsu Province Natural Science Foundation of China under Grant No. BK2009058. Also supported by NDC Lab, KICT, IIUM, Malaysia.

References

[1] Z. Liu, P. Dhungel, D. Wu, C. Zhang and K. W. Ross, Understanding and Improving Incentives in Private P2P Communities. In ICDCS 2010, Italy, Jun. 2010.

[2] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, The EigenTrust Algorithm for Reputation Management in P2P Networks. In Proceedings of the Twelfth International World Wide Web Conference, Budapest, May 2003.

[3] Q. Lian, Z. Zhang, M. Yang, B. Y. Zhao, Dai Y. and X. Li, An empirical study of collusion behavior in the maze P2P file-sharing system. In IEEE ICDCS, June 2007.

[4] J. R. Douceur, The Sybil attack. In First International Workshop on Peer-to-Peer Systems (IPTPS ’02), Mar. 2002.

[5] L. Mekouar, Y. Iraqi and R. Boutaba, Peer-to-Peer's Most Wanted: Malicious Peers. Comp. Net., vol. 50, no. 4, Mar. 2006, pp. 545–62.

[6] W. Ji, S. Yang and B. Chen, A Group-Based Trust Metric for P2P Networks: Protection against Sybil Attack and Collusion. International Conference on Computer Science and Software Engineering, 2008 Volume: 3, Page(s): 90 - 93.

[7] H. Lee, J. Kim and K. Shin, Simplified clique detection for collusion-resistant reputation management scheme in P2P networks. In 2010 International Symposium on Communications and Information Technologies (ISCIT), 2010, Page(s): 273 - 278.

[8] Z. Despotovic and K. Aberer, P2P reputation management: Probabilistic estimation vs. social networks, Computer Networks 50 (2006) 485–500.

[9] G. Ciccarelli and R. L. Cigno, Collusion in peer-to-peer systems, Computer Networks, Comp. Net., vol. 55, no. 15, Oct. 2011, pp. 3517–3532.

[10] Y. Liu, Y. Li, N. Xiong, J. H. Park and Y. S. Lee, The incentive secure mechanism based on quality of service in P2P network, Computers and Mathematics with Applications, 60 (2010) 224-233.

[11] X. Wei, T. Ahmed, M. Chen, and A.-S.K. Pathan, PeerMate: A malicious peer detection algorithm for P2P Systems based on MSPCA, in Proc. IEEE Int. Conf. on Computing, Networking and Communications (ICNC), Lahaina, HI, USA, Jan. 2012, pp.815-819..

[12] M. Meulpolder, J.A. Pouwelse, D.H.J. Epema, and H.J. Sips, BarterCast: A practical approach to prevent lazy freeriding in P2P networks. in Proc. IEEE International Symposium on Parallel & Distributed Processing (IPDPS'09), 2009, pp. 1-8.

[13] H. Jiang, H. Jin, S. Guo, and X. Liao, A measurement-based study on user management in private BitTorrent communities. Concurrency and Computation: Practice and Experience, DOI: 10.1002/cpe.2884, 2012.

[14] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy and J. Zahorjan, Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In 19-th ACM Symposium on Operating Systems Principles, Bolton Landing, NY, USA, October 2003.

[15] X. Wei, M. Chen, G. Tang, H. Bai, G. Zhang and Z. Wang. iRep: Indirect Reciprocity Reputation based Efficient Content Delivery in BT-like Systems. Telecommunication Systems, to be appear.

[16] R. Landa, D. Griffin, R. Clegg, E. Mykoniati, and M. Rio, A sybilproof indirect reciprocity mechanism for peer-to-peer networks, In Proceedings of IEEE Infocom' 09, 2009.

[17] K. Hoffman, D. Zage, and C. Nita-Rotaru, A survey of attack and defense techniques for reputation systems. Technical Report of Purdue University (CSD TR No.07-013), 2007.

[18] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, Clustering and sharing incentives in BitTorrent systems, In SIGMETRICS Perform. Eval. Rev., 2007.
[19] A. Al-Hamra, A. Legout, and C. Barakat, Understanding the properties of the bittorrent overlay, INRIA, Tech. Rep., 2007. [Online]. Available: http://arxiv.org/pdf/0707.1820.
[20] A. Lakhina, M. Crovella and C. Diot, Diagnosing Network-Wide Traffic Anomalies, in Proc. SIGCOMM, Portland, Oregon, USA, 2004.
[21] T. Ahmed, M. Coates, and A. Lakhina, Multivariate online anomaly detection using kernel recursive least squares, in Proc. IEEE INFOCOM, Anchorage, AK, May 2007.
[22] T. Ahmed, X. Wei, S. Ahmed and A.-S.K. Pathan, Intruder Detection in Camera Networks Using the One-Class Neighbor Machine, in Proc. Networking and Electronic Commerce Research Conference (NAEC), Riva Del Garda, Italy, Oct 2011.
[23] D. C. Montgomery, Introduction to Statistical Quality Control, Second Edition, New York: John Wiley & Sons, Inc. 1991.
[24] B. R. Bakshi, Multiscale PCA with Application to Multivariate Statistical Process Monitoring, AIChE journal, 1998, 44(3): 1596-1610.
[25] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, et al., Wavelet Shrinkage: Asymptopia ?, J. R. Stat. Soc. B, 57, 2797-2814, 1995.
[26] D. L. Donoho, De-noising by soft-thresholding, IEEE Trans. Inform. Theory, vol. 41, no. 3, pp. 613–627, May 1995.
[27] Y. Yang, Q. Feng, Y. L. Sun, and Y. Dai. 2008. RepTrap: a novel attack on feedback-based reputation systems. In Proceedings of the 4th international conference on Security and privacy in communication networks (SecureComm '08). New York, NY, USA.
[28] A. Tehale, A. Sadafule, S. Shirsat, R. Jadhav, S. Umbarje, and S. Shingade. Parental Control algorithm for Sybil detection in distributed P2P networks. International Journal of Scientific and Research Publications, Volume 2, Issue 5, May 2012.
[29] C. Selvaraj and S. Anand. A survey on Security Issues of Reputation Management Systems for Peer-to-Peer Networks. Computer Science Review, In press.
[30] X. Jin and S.-H. G. Chan. Detecting malicious nodes in peer-to-peer streaming by peer-based monitoring. ACM Trans. Multimedia Comput. Commun. Appl., vol. 6, no. 2, pp. 9:1–9:18, 2010.
[31] E. Koutrouli and A. Tsalgatidou. Taxonomy of attacks and defense mechanisms in P2P reputation systems—Lessons for reputation system designers. Computer Science Review, 2012.