Computer Networks 108 (2016) 29–39

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

An outlier mining-based malicious node detection model for hybrid P2P networks

Xianfu Meng*, Shuang Ren

School of Computer Science and Technology, Dalian University of Technology, No. 2, Linggong Road, Dalian 116024, China

ARTICLE INFO

Article history:
Received 30 October 2015
Revised 5 May 2016
Accepted 22 July 2016
Available online 30 July 2016

Keywords:
P2P network
Malicious node detection
Behavior pattern
Frequent behavior pattern
Outlier mining

ABSTRACT

With the increases of P2P applications and their users, the malicious attacks also increased significantly, which negatively impacts on the availability of the P2P networks and their users' experience. This paper presents an outlier mining-based malicious node detection model for hybrid P2P networks. We first extract the local nodes' frequent patterns from the nodes' behavior patterns in subnets using the frequent behavior pattern mining approach, and then we produce and update the nodes' global frequent behavior patterns by incrementally propagating and aggregating the local frequent behavior patterns. Finally, we identify outliers (i.e. the malicious nodes) using the local frequent behavior patterns and the global frequent behavior patterns. We also discuss how to recognize the different types of malicious nodes from outliers. Simulation results show that our strategy could detect malicious nodes with low false positive rate and low false negative rate.

1. Introduction

To strengthen the reliability and security of P2P networks, researchers have proposed a lot of approaches, in which the feedback-based trust models take the major part [1–10]. In a feedback-based trust model, a peer's credibility is computed by using the feedbacks on the peer's past services. However, the existence of a lot of false feedbacks makes such trust models unable to effectively and efficiently curb malicious behavior, such as the collusion attacks and the Sybil attacks. Though we could use the false feedback filtering mechanisms in such trust models, since some of the authentic feedbacks are also screened out at the same time, the effectiveness of the trust models cannot be guaranteed [8]. Moreover, the feedback-based trust models usually set a trust or weight for each peer [6], which is used to determine how credible the peer is, without identifying what category the peer belongs to. Hence, it is difficult for such trust models to effectively and efficiently curb malicious behavior. To tackle the problems, researchers have proposed the trust-based malicious peer separation approaches, PeerMate [11] and SMART [12]. However, since a peer's trust is calculated by using globally collected feedbacks, these two algorithms suffer from the same shortcomings as those mentioned above. Particularly, since the two approaches only use two values of 0 and 1 as the detection results, which make it difficult to further identify the types of the malicious peers.

In a hybrid P2P network, all the nodes are classified into two categories, one is the super node and another is the ordinary node. Under each super node, there are several ordinary nodes with which the super node forms a subnet. Each super node stores its neighbor super nodes' list, so as to ease the communications between subnets. In a subnet, the super node is responsible for managing the interaction data among the ordinary peers. Meanwhile, all the super nodes are in charge of managing the interaction data among the subnets. Based on the finding that each malicious peer has the specific characteristic of outlier [1], this paper proposes a malicious node detection model using outlier mining approach in the hybrid P2P networks. Our main work is the following.

(1) We make use of the interaction data kept in the super nodes to detect malicious nodes, which could eliminate the false feedback problem existed in the feedback-based P2P trust models.
(2) We extract local frequent behavior patterns for each subnet, and achieve the global frequent behavior patterns by incrementally propagating and aggregating the local frequent behavior patterns coming from the super nodes, by which the outliers (i.e. the malicious nodes) could be detected.
(3) We present the algorithm of recognizing different types of malicious nodes based on the local frequent behavior patterns and the global frequent behavior patterns, which could help peers curb malicious attacks better.

The simulation results showed that our approach outperforms the models of EigenTrust [2], PeerMate [11] and SMART [12] in terms of the false positive rate and the false negative rate.

* Corresponding author.
E-mail address: xfmeng@dlut.edu.cn (X. Meng).

http://dx.doi.org/10.1016/j.comnet.2016.07.008
1389-1286/

The remainder of the paper is organized as follows. Section 2 presents the related work on the P2P trust models and the P2P malicious node detection approaches. Section 3 presents the network structure and the definitions. The frequent behavior pattern mining-based malicious peer detection model is detailed in Section 4, and the examples of applying our model to detect collusion, Sybil and file pollution attacks are described in Section 5. Section 6 evaluates our model's effectiveness with simulations. Finally, Section 7 concludes the paper and gives our future research focus.

2. Related work

P2P trust models can be classified into the local trust model and the global trust model. The former usually uses the locally collected feedback information to calculate a peer's credibility [2], and the latter computes a peer's credibility mainly by aggregating the direct trust owned by the evaluation peer and the indirect trust calculated by using the globally collected feedbacks [1–5]. EigenTrust [2] is a typical global trust model. In EigenTrust, every peer i owns a unique global trust, and any peer which has received services from peer i holds its local trust. This model aggregates all the peers' local trust to calculate peer i's unique global trust. However, when applied to a large scale network, EigenTrust suffers from the problems of bad scalability, slow convergence and high computational complexity. Furthermore, EigenTrust assumes the existence of some pre-trusted peers, which is an obstacle to applying the trust model in the real world P2P networks. Douwen [5] presented the global trust model based on feedbacks, expecting to eliminate the convergence problem of trust iteration existed in EigenTrust. However, since its effectiveness relies heavily on the feedbacks, this strategy could not work well under the situation that there are false feedbacks or insufficient feedbacks. Meanwhile, such feedback-based trust models would in turn incur the malicious attacks, such as collusion attack [1,6], Sybil attack [7] and false feedback [8].

The existing approaches used to resist malicious behavior can be classified into two types. The first type is to set a trust or weight for each peer [6], which is used to determine how to treat the peer, without identifying what category the peer belongs to. Because this type of approach could not identify the categories of malicious peers, it is difficult to effectively and efficiently curb malicious behavior. Most existing trust models belong to this type. The second type is that it could identify the categories of malicious behavior. This type of approach could further be divided into two kinds according to what detection method it uses. In [1,6] and [9], the authors used the bottom-up method to detect malicious behavior, which requires a certain amount of prior knowledge or experience to sum up the conditions of judging malicious behavior. The peers satisfying a set of given conditions are taken to be the malicious peers. Lian et al. [1] derives the characteristics of collusion behavior from analyzing the users' log file, and based on which to identify collusion attacks. It has also concluded that the peers which behave collusively are outliers. This kind of method is effective in curbing specific malicious attack, but it is difficult to use such method to resist complex or mixed attacks. Mekouar et al. [9] detects the peers of providing false feedbacks by comparing the difference of the feedbacks given by the two sides of a transaction. If the difference is significant, the two peers are considered to be suspicious in providing false feedbacks. In [8,10] and [11], the authors used the top-down method to curb malicious behavior. This kind of approach first distinguishes abnormal peers and then analyzes and curbs the malicious peers. Cai et al. [10] presented a collusion detection trust model based on peers' behavior similarity. The model assumes that the peers in a collusive group have the similar behavior patterns, and if the number of peers in a group is big enough and the similarity of the peers' behavior is greater than a threshold, then the group is taken to be the collusive group. However, how to determine the threshold is a difficult task. PeerMate [11] used Multiscale Principal Component Analysis (MSPCA) method to separate malicious peers. However, it could not identify a part of Sybil nodes which have the similar behavior to the normal peers, leading to the problem that some of the normal peers are mistakenly identified as the malicious peers. Wei et al. [12] made an improvement on PeerMate, and proposed the algorithm of SMART. Though SMART could separate Sybil peers, not a few normal peers are mistakenly identified as the malicious ones. Since these two algorithms are both based on peers' global trust which is calculated by using the globally collected feedbacks, the correctness of the computed peers' trust is difficult to be guaranteed under malicious attacks. Particularly, the two strategies only use two values of 1 and 0 as the detection results, which are hard to be used to further identify the types of the malicious peers and thus negatively impact on the effectiveness of curbing malicious behavior.

Li et al. [13] proposed a distributed detection model for curbing malicious peers. Each peer is responsible for detecting its neighbor peers. Initially, all the peers are in the indefinite peer set, which means each peer is indefinite on whether it is malicious or not. A peer determines its neighbor peer's status based on the determination of the status of the peer's neighbor peer set. By performing iterative operations, this strategy could gradually move the peers in the indefinite peer set to the normal peer set and the malicious peer set. However, it suffers from the problem of high computational complexity due to the iterative operations.

Outlier mining is a hot research topic in data mining area, which is mainly used in noise elimination and knowledge discovery in database, as well as fraud detection and intrusion detection in network. Outlier detection can be classified into several classes, such as classification-based, depth-based, cluster-based, distance-based and density-based [14]. He et al. [15] presented the frequent pattern based outlier detection approach, and pointed out that the discovered frequent patterns reflect the "common features" in dataset. In other words, if a data object contains more frequent patterns, then it means this data object is unlikely to be an outlier because it possesses the "common features" of the dataset. However, this algorithm neglects the impact of the length of the frequent patterns on the evaluation of the outlier factor, and also it has higher computational complexity. Zhou et al. [16] made an improvement on the frequent pattern based outlier detection approach, which makes use of the weight of frequent patterns to determine the importance of different frequent patterns in finding outliers.

Agrawal et al. [17] proposed the concept of frequent pattern mining for the first time, aiming at analyzing the market basket with association rule mining. Apriori [18] computed frequent item sets by using layer-wise iterative operations, based on which to obtain the association rules. It produces a lot of candidate item sets, and meanwhile needs to scan database many times. Therefore, it suffers from the problems of high computational complexity and bad spatial scalability. To tackle the problems, Han et al. [19] proposed the algorithm of FT-growth, which could directly generate the frequent patterns without producing the candidate item sets. However, the above mentioned frequent pattern mining algorithms require the users to set the parameter of min_*support*, the minimum support threshold which is used for choosing the frequent patterns. This is an obstacle to using such algorithms. To avoid the problem, AbdusSalam et al. [20] proposed a top-k mining algorithm, which needs not set the parameter of min_*support*. Also, it only needs to scan the database once and produces 2-item sets. Such merits make it suitable to be applied to many fields, such as outlier mining in P2P networks.

3. Descriptions of network structure and related definitions

In this section, we describe the network structure on which our model is implemented, and we also present the related definitions. In the following descriptions, we will use the words of peer and node interchangeably.

3.1. The network structure

To effectively and sufficiently make use of the peers' heterogeneity, we implement our model on a hybrid P2P network. All the peers in the network are classified into two layers, one layer consists of super peers and another layer is with ordinary peers. Under each super peer, there are several ordinary peers with which the super peer forms a subnet. Each super node maintains its neighbor super nodes' list, so as to ease the communications between subnets. There are two things each super peer should do in our model. In a subnet, the super node is responsible for managing the interaction data among the ordinary peers to establish ordinary peers' behavior patterns, mine the local frequent behavior patterns and calculate the outlier factors used to detect malicious peers. Among the subnets, each super peer is in charge of propagating and aggregating messages related to each subnet's frequent behavior patterns and thus gradually establishes the global frequent behavior patterns.

Our model could be directly applied to the existing hybrid P2P networks, such as the networks mentioned in [21–25]. In these networks, the super peers (e.g. the SuperNodes in KaZaA [21], the servers in eDonkey [22] and the UltraPeers in Gnutella 0.6 [23]) are usually composed of the peers with stronger processing capacity [22,24], high bandwidth [22,25], huge storage [24,25] and long online time [26]. Also, all the super peers construct an overlay network. As for the network with no super peers, we could first construct a hybrid P2P network with super peers, and then apply our model to curb malicious behavior.

In [24] and [25], the authors pointed out that the super peers in a hybrid P2P network usually consist of the peers with high reliability since they play the important roles in data management and control. Without loss of generality, this paper assumes that all the super peers are credible and reliable, which means they behave honestly and normally in calculations, managements and communications without making malicious attacks on other peers. If we want to use an ordinary peer as the super peer, we can choose the peer with high credibility as the super peer by using the algorithms proposed in [24,25].

3.2. Related definitions

Definition 1. (Behavior pattern). A peer's behavior pattern, denoted by *BP*, refers to the ordered key-value set derived from the peer's interaction data with other peers, which is used to quantify the peer's behavioral manner.

For any peer i, its *BP* is represented by $BP_i=\{(^{I_1}_{v_1^i}),(^{I_2}_{v_2^i}),\cdots,(^{I_s}_{v_s^i})\}$, where $I=\{I_1, I_2,\cdots, I_s\}$ stands for the set of keys reflecting the features of peer i's interaction behavior, and v_j^i $(1\le j\le s)$ is the value of key I_j. An arbitrary non-empty subset of BP_i is called a behavior pattern of peer i. For example,$\{(^{I_1}_{v_1^i})\}$ and$\{(^{I_1}_{v_1^i}),(^{I_3}_{v_3^i})\}$ are both peer i's behavior patterns. *BP*'s length refers to the number of items in *BP*, denoted by $||BP||$. Note that the behavior patterns mentioned here are called the initial behavior patterns which are not directly used for the malicious peer detection.

We take the P2P file sharing network as an example to set the value of *BP* for detecting collusion peers. According to [1], the set of keys can be set to I={duplication degree, pair-wise degree, PM ratio, traffic concentration degree}, where *duplication degree* is the ratio of total upload traffic (bytes) to the amount of non-duplicated data (bytes); *pair-wise degree* is the ratio of total traffic between two peers to the sum of all traffic uploaded by both peers; *PM ratio* is the ratio of number of peers to the number of machines to describe how densely a peer's clients are distributed across different physical machines; *traffic concentration degree* is the ratio of a peer's highest upload traffic to a single machine to its total upload traffic. We assume that peer i takes the values of 10, 0.87, 1 and 0.93 in each key respectively, then peer i's initial behavior pattern can be represented by $BP_i=\{(^{I_1}_{10}),(^{I_2}_{0.87}),(^{I_3}_{1}),(^{I_4}_{0.93})\}$. The non-empty subsets of BP_i, such as $\{(^{I_1}_{10}),(^{I_2}_{0.87})\}$ and $\{(^{I_3}_{1})\}$, are the initial behavior patterns to which peer i conforms, and their length is respectively 2 and 1.

Definition 2. (Local database). The dataset of peers' behavior patterns stored in the super node (SN) of a subnet is called the local database of the subnet, denoted by D_{SN}.

Definition 3. (Global database). All the local databases managed by the super peers construct the global database, denoted by D.

Definition 4. (Frequent behavior pattern). Frequent behavior pattern refers to the behavior pattern frequently occurred in a peer's behavior pattern set, denoted by *FP*.

Definition 5. (Local frequent behavior pattern). Local frequent behavior pattern refers to the frequent behavior pattern occurred in local database, denoted by LocalFP.

Definition 6. (Global frequent behavior pattern). Global frequent behavior pattern refers to the frequent behavior pattern occurred in global database, denoted by GlobalFP.

4. Frequent behavior pattern mining-based malicious peer detection model

4.1. Peers' behavior patterns and their management

Each super node takes T as the period to establish peers' local behavior patterns, and saves all the *BP* data within time window τ. Here, $\tau=\{T_1, T_2, \cdots, T_m\}$, and m is the number of periods a time window τ consists of. The local database of the subnet super peer SN is in can be represented by $D_{SN}=[D_{SN}^{T_1}, D_{SN}^{T_2},\cdots, D_{SN}^{T_m}]$, where $T_1 \sim T_m$ are the consecutive periods in time window τ, and $D_{SN}^{T_x}(1\le x\le m)$ is the *BP* set of the subnet super peer SN is in within period T_x.

In the end of any period T_x within time window τ, the super peer SN creates $D_{SN}^{T_x}$ based on its managed peers' interaction data. The procedure of creating $D_{SN}^{T_x}$ is as follows.

(1) Data pre-processing

To guarantee the correctness and integrity of the interaction data, this step is used to eliminate the incomplete and improperly formatted interaction data.

(2) Calculation of the initial behavior patterns

Super peer SN calculates the value of each key of all the peers in the subnet, and obtains the local data of $D_{SN}^{T_x}$ as follows.

$$D_{SN}^{T_x}=\begin{pmatrix} BP_1^{T_x}\\ BP_2^{T_x}\\ \vdots\\ BP_n^{T_x}\end{pmatrix}=\begin{pmatrix} v_1^{1,T_x} & v_2^{1,T_x} & \cdots & v_s^{1,T_x}\\ v_1^{2,T_x} & v_2^{2,T_x} & \cdots & v_s^{2,T_x}\\ \vdots & \vdots & & \vdots\\ v_1^{n,T_x} & v_2^{n,T_x} & \cdots & v_s^{n,T_x}\end{pmatrix} \quad (1)$$

where v_j^{i,T_x} $(1 \le j \le s,\ 1 \le i \le n)$ is the value of key I_j peer i gets in period T_x.

(3) Normalization of the initial behavior patterns

Before mining outliers, we first normalize peers' initial *BP* data, so as to ease the mining and analyzing process. Suppose the range of the kth column (i.e. the kth key) in $D_{SN}^{T_x}$ is [min, max] and we want to divide the range into h intervals. Then, the length of each interval is d=(max-min)/h, and thus the range of each interval is [min, min+d), [min+d, min+2d), ..., [min+(h-1)d, max]. We use 0 for the data in the first interval of [min, min+d), 1 for the data in the second interval of [min+d, min+2d), ..., h-1 for the data in the last interval of [min+(h-1)d, max] to recalculate $D_{SN}^{T_x}$, as shown in Formula (2).

$$D_{SN}^{T_x} = \begin{pmatrix} BP_1^{T_x} \\ BP_2^{T_x} \\ \vdots \\ BP_n^{T_x} \end{pmatrix} = \begin{pmatrix} C_1^{1,T_x} & C_2^{1,T_x} & \cdots & C_s^{1,T_x} \\ C_1^{2,T_x} & C_2^{2,T_x} & \cdots & C_s^{2,T_x} \\ \vdots & \vdots & & \vdots \\ C_1^{n,T_x} & C_2^{n,T_x} & \cdots & C_s^{n,T_x} \end{pmatrix} \tag{2}$$

where C_j^{i,T_x} $(1 \le j \le s,\ 1 \le i \le n)$ is the number of interval v_j^{i,T_x} belongs to.

4.2. FP mining process with a distributed approach

To curb malicious peers, we should first mine the normal peers' behavior patterns, and based on which to find outliers. In this section, we detail the mining processes of LocalFP and GlobalFP.

4.2.1. The mining process of LocalFP

(1) Conditions of triggering the mining of LocalFP

To reduce the complexity, we first need to determine when the LocalFP mining should be triggered. When the first time window elapsed, we trigger the mining of LocalFP to obtain the initial local frequent behavior patterns of each subnet. In the end of subsequent time window, we compare peers' *BPs* of the current time window with those of the previous time window to determine whether to trigger the mining of LocalFP or not.

Let $\overline{BP_i^\tau}$ represent the average value of peer i's *BP* in time window τ, $\overline{D_{SN}^\tau}$ stand for the average value of D_{SN} in time window τ. Then, $\overline{D_{SN}^\tau}$ can be calculated with Formula (3).

$$\overline{D_{SN}^\tau} = \begin{pmatrix} \overline{BP_1^\tau} \\ \overline{BP_2^\tau} \\ \vdots \\ \overline{BP_n^\tau} \end{pmatrix} = \begin{pmatrix} \overline{C_1^{1,\tau}} & \overline{C_2^{1,\tau}} & \cdots & \overline{C_s^{1,\tau}} \\ \overline{C_1^{2,\tau}} & \overline{C_2^{2,\tau}} & \cdots & \overline{C_s^{2,\tau}} \\ \vdots & \vdots & & \vdots \\ \overline{C_1^{n,\tau}} & \overline{C_2^{n,\tau}} & \cdots & \overline{C_s^{n,\tau}} \end{pmatrix} \tag{3}$$

where $\overline{C_j^{i,\tau}} = \frac{1}{m}\sum_{x=1}^{m} C_j^{i,T_x}$ and m is the number of periods time window τ has. Here, C_j^{i,T_x} $(1 \le j \le s,\ 1 \le i \le n)$ is the same as shown in Formula (2).

Based on the above calculations, the following two conditions are used to trigger the mining of LocalFP.

1) The total variation of peers' FP-based triggering of mining LocalFP

Let τ be the current time window, τ' be the τ's previous time window, $V_{SN}(\tau,\ \tau')$ be the variation of $\overline{D_{SN}^\tau}$ and $\overline{D_{SN}^{\tau'}}$. Then, $V_{SN}(\tau,\ \tau')$ is calculated as follows.

$$V_{SN}(\tau,\ \tau') = \frac{1}{n}\sum_{i=1}^{n} \rho\left(\overline{BP_i^\tau}, \overline{BP_i^{\tau'}}\right) \tag{4}$$

where n is the number of peers in the subnet super peer SN is in, $\rho(\overline{BP_i^\tau}, \overline{BP_i^{\tau'}})$is the variation of $\overline{BP_i^\tau}$ and $\overline{BP_i^{\tau'}}$, as defined below.

$$\rho\left(\overline{BP_i^\tau}, \overline{BP_i^{\tau'}}\right) = \sqrt{\sum_{j=1}^{s}\left(\overline{C_j^{i,\tau}} - \overline{C_j^{i,\tau'}}\right)^2} \tag{5}$$

where s=||I|| is the number of keys, $\overline{C_j^{i,\tau}}$ $(1 \le j \le s,\ 1 \le i \le n)$ is the same as mentioned in Formula (3).

When $V_{SN}(\tau,\ \tau')$ is greater than threshold μ_1, the mining process of LocalFP should be triggered.

2) The variation of individual peer's FP-based triggering of mining LocalFP

Let τ represent the current time window, τ' stand for the previous time window of τ. Let $MaxV_{SN}(\tau,\ \tau')$ represent the maximum variation of all peers' FPs in time windows τ and τ'. Then, $MaxV_{SN}(\tau,\ \tau')$ can be calculated as follows.

$$MaxV_{SN}(\tau,\ \tau') = Max\left\{\rho\left(\overline{BP_i^\tau}, \overline{BP_i^{\tau'}}\right) |\ 1 \le i \le n\right\} \tag{6}$$

When $MaxV_{SN}(\tau,\ \tau')$ is greater than threshold μ_2, the mining process of LocalFP should also be triggered.

(2) The mining process of LocalFP

Frequent behavior pattern mining is a hot research topic in data mining area, and a lot of algorithms have been proposed. In this paper, we adopt the algorithm presented in [20] to complete mining LocalFP, since this algorithm only needs to scan the database once and needs not set the parameter of min_*support* when mining top-k frequent behavior patterns. We briefly list the steps of this algorithm as follows.

1) Scan local database to produce all the 2-item set and save them to matrix M, in which the value of each 2-item is the number (i.e. the frequency) of the 2-items existed in the local database.
2) Change the frequencies of 2-item set in M to *AR* (association ratio) values, where $AR(i,\ j) = P(x_i, x_j)/(1 - P(x_i, x_j))$ and $P(x_i, x_j)$ is the probability of the frequency that x_i and x_j occur simultaneously.
3) Arrange the 2-item set in M in the descent order of *AR* values, and save the ordered result into list L.
4) Create *AR* graph from L, and produce the most frequent behavior patterns by using ASD-tree (all-path-source-destination tree) according to AR graph. When the k most frequent behavior patterns have been produced, the process ends.

To enhance the portability of our model, we take the FP mining algorithm as a black box. So, we can use any existing FP mining algorithm, such as Apriori [18], FPgrowth [19], and so on.

When the LocalFP mining process is triggered in the super node of a subnet, we use Update.inc to represent the newly added FPs and Update.del to stand for the BPs which are no longer frequent patterns based on the comparison between the newly achieved LocalFPs and the old ones. Update.inc and Update.del reflect the change of LocalFPs and will be used to incrementally update the GlobalFPs as described in Section 4.2.2.

Based on the above descriptions, we present the local frequent behavior pattern mining algorithm in Algorithm 1.

4.2.2. Establishment of GlobalFPs

The local frequent behavior patterns only reflect the peers' behavior in a subnet, which could not reflect the features of peers' behavior in the overall P2P network. If a subnet is controlled by a collusion group, we could not use the local frequent behavior patterns to identify the malicious nodes. In such situation, we should

Table 1
Definitions of message structure.

Sender's address	Super node's IP address
Update.Inc	Update.Inc : $\{(FP_1, IF), (FP_2, IF), \cdots, (FP_{N_{Inc}}, IF)\}$
Update.Del	Update.Del : $\{(FP_1, IF), (FP_2, IF), \cdots, (FP_{N_{Del}}, IF)\}$

make use of the global frequent behavior patterns to remedy the flaw.

GlobalFPs are established by incrementally propagating and aggregating the LocalFPs among the super nodes. This process is completed by using the messages propagated among the super nodes, as described below.

(1) Message structure definition used for incrementally propagating LocalFPs

Initially, the LocalFPs of a subnet are the GlobalFPs of the subnet, which are managed by the super node of the subnet. When the super node of a subnet completed the mining process of LocalFPs, the super node would send a message with Update.inc and Update.del to its neighboring super nodes, the neighboring super nodes continue this process until the message reaches all the super nodes. By this process, the GlobalFPs in each subnet are updated, which could be used for identifying malicious peers.

When the super node of a subnet creates Update.inc and Update.del, it adds an impact factor to each *FP* in Update.inc and Update.del, which reflects the importance of the *FP* for evaluating the outliers. The impact factor of an *FP* is calculated as follows.

$$IF(FP) = \frac{1}{s} \times \|FP\| \times subNetSize(SN) \tag{7}$$

where s=||*I*||, and *subNetSize(SN)* is the number of peers in the subnet super peer *SN* is in. From Formula (7), we see that the more the number of peers in the subnet is and the more the number of items in an *FP* is, the higher the value of the impact factor of the *FP* is. This is because the higher number of peers in a subnet and the higher number of items in an *FP* indicate the higher importance of such *FP* for reflecting the normal peers' behavior features.

In order to incrementally propagating LocalFPs among the super nodes, we define the message structure as follows.

In Table 1, FP_i is the frequent behavior pattern, and *IF* is the impact factor of the corresponding *FP*.

Each super node is both the sender and the receiver of the messages. When a super node has received the messages more than threshold δ, it initiates the update operation of GlobalFPs. In the initial phase, the update should be frequent to construct GlobalFPs, and thus the δ should be set to a smaller value. As the time goes on, the δ should be set to a higher value, since the GlobalFPs in each super node tend to be no significant difference from each other. In such situation, it is no need to frequently update GlobalFPs. Based on this consideration, we calculate threshold δ as follows.

$$\delta_t = \left(1 - e^{-(1+t-t_0)}\right) \times N \tag{8}$$

where N is the number of super peers in the network, t_0 is the time at which the super node received the first message.

(2) The update of GlobalFPs

The update process of GlobalFPs is as follows.

1) Message aggregation

Let Q_{SN} represent the message set received by super node *SN*. If there exist multiple records of an *FP* in Q_{SN}, we aggregate these records to produce one record for the *FP*. In other words, we only keep one record for each *FP* in Q_{SN}. As shown in Table 1, each *FP* has an *IF* with it no matter it is in Update.inc or in Update.del. When aggregating the records of an *FP*, we only need summing up their *IFs*. We add the value of *IF* whose corresponding *FP* is in Update.inc and deduct the value of *IF* whose corresponding *FP* is in Update.del. If the calculated value of *IF*, say IF_c, is greater than zero, we keep the record of *FP* in Update.inc whose corresponding *IF* is set to IF_c; if IF_c<0, we keep the record of *FP* in Update.del whose corresponding *IF* is set to abs (IF_c); if IF_c=0; we keep no record for the *FP*. Details can be found in Algorithm 2.

2) The update of GlobalFP

To ease the descriptions, we first describe the meaning of some notations.

Let Q_{SN}(Update.inc) represent the set of Update.inc in Q_{SN}, and Q_{SN}(Update.del) stand for the set of Update.del in Q_{SN}. Let Q_{SN}(Update.inc).FP represent an *FP* in Q_{SN}(Update.inc), and Q_{SN}(Update.inc).FP->IF stand for the *IF* whose corresponding *FP* is in Q_{SN}(Update.inc). Let Q_{SN}(Update.del). FP represent an *FP* in Q_{SN}(Update.del), and Q_{SN}(Update.del).FP->IF stand for the *IF* whose corresponding *FP* is in Q_{SN}(Update.del). Let GlobalFPs.FP represent the *FP* in GlobalFPs, and GlobalFPs.FP->IF stand for the *IF* whose corresponding *FP* is in GlobalFPs.

When the number of *FPs* in Q_{SN} is greater than threshold δ in a subnet, we start updating GlobalFPs as shown in Algorithm 2.

In Algorithm 2, we take the delay of message transmissions among the super nodes into account to treat the *FPs* in Q_{SN}(Update.del), so as to ensure the consistency of GlobalFPs in each super node.

4.3. LocalFP and globalfps-based malicious node detection

We have discussed the establishment and update of LocalFPs and GlobalFPs. Generally speaking, if a node has fewer *BPs* consistent with the *FPs*, then the possibility that the node is an outlier and thus a malicious node is high [15,16]. Based on this finding, to verify whether a node is malicious or not, we first calculate the node's outlier factor, including the local outlier factor (LocalOF) and the global outlier factor (GlobalOF), by using LocalFPs and GlobalFPs. Any node *i*'s LocalOF and GlobalOF are calculated as follows.

$$LocalOF_i = 1 - \frac{\sum_{X \subseteq BP_i, X \in LocalFP_{SN}} w(X)}{\|LocalFP_{SN}\|} \tag{9}$$

where *X* is a local *FP* to which node *i*'s *BPs* conform, $w(X) = \frac{1}{s} \times \|X\|$ is the weight of *X*, and s=||*I*|| is the number of keys.

From Formula (9), we see that the fewer a node's *BPs* conforming to the Local *FPs* are, the greater its LocalOF is and thus the higher the possibility that the node is an outlier is.

$$GlobalOF_i = 1 - \frac{\sum_{X \subseteq BP_i, X \in GlobalFP_{SN}} IF(X)}{\sum_{Y \in GlobalFP_{SN}} IF(Y)} \tag{10}$$

where *X* is a global *FP* to which node *i*'s *BPs* conform, and *IF(X)* is *X*'s impact factor as defined in (7).

From Formula (10), we see that the fewer a node's *BPs* conforming to the global *FPs* are, the greater its GlobalOF is and thus the higher the possibility that the node is an outlier is.

Let $\overline{LocalOF_{SN}}$ and $\overline{GlobalOF_{SN}}$ represent the average local outlier factor and the average global outlier factor of the subnet with super node *SN*. Then, $\overline{LocalOF_{SN}}$ and $\overline{GlobalOF_{SN}}$ are calculated as follows.

$$\overline{LocalOF_{SN}} = \frac{1}{subNetSize(SN)} \sum_{i \in subNet(SN)} LocalOF_i \tag{11}$$

$$\overline{GlobalOF_{SN}} = \frac{1}{subNetSize(SN)} \sum_{i \in subNet(SN)} GlobalOF_i \tag{12}$$

where *subNet(SN)* is the node set of the subnet with super node *SN*, and *subNetSize*(*SN*) is the number of nodes in the subnet with super node *SN*.

According to the existing researches [2,4,8,11,12], most nodes in a P2P network are normal and well-meant ones. Based on this finding, we present the malicious node detection approach as follows.

(1) For any node *i* in the subnet with super node *SN*, if $LocalOF_i \le \overline{LocalOF_{SN}}$ and $GlobalOF_i \le \overline{GlobalOF_{SN}}$, then it means that node *i*'s *BPs* conform to both the local *FPs* and global *FPs*. Thus, node *i* is a normal node and the subnet node *i* is in is a normal subnet.
(2) For any node *i* in the subnet with super node *SN*, if $LocalOF_i > \overline{LocalOF_{SN}}$ and $GlobalOF_i \le \overline{GlobalOF_{SN}}$, then it means that node *i* is an outlier in its subnet but not an outlier in the global network. In other words, node *i*'s *BPs* conform to the global *FPs*, but are not consistent with the local *FPs*. Thus, node *i* is a normal node, but the subnet node *i* is in is an abnormal subnet.
(3) For any node *i* in the subnet with super node *SN*, if $LocalOF_i \le \overline{LocalOF_{SN}}$ and $GlobalOF_i > \overline{GlobalOF_{SN}}$, then it means node *i* is not an outlier in its subnet, but an outlier in the global network. In other words, node *i*'s *BPs* conform to the local *FPs*, but are not consistent with the global *FPs*. Therefore, node *i* is a malicious node and the subnet node *i* is in is an abnormal subnet.
(4) For any node *i* in the subnet with super node *SN*, if $LocalOF_i > \overline{LocalOF_{SN}}$ and $GlobalOF_i > \overline{GlobalOF_{SN}}$, then node *i* is an outlier in both the subnet and the global network. In such situation, node *i* is a malicious node, and the subnet node *i* is in might be a normal or an abnormal subnet.

For an abnormal subnet as shown in case (3), it might be a collusion group. In such case, the *BPs* existed in LocalFPs but not in GlobalFPs are considered to be the behavioral features of the collusion group. Based on this, the super node of the subnet could curb the malicious behavior and notify other super nodes. For a malicious peer, if its *BPs* do not conform to the *BPs* of any collusion group, then it is an individual malicious peer.

5. Examples of applying our model to curb malicious attacks

In this section, we take the examples of curbing collusion, Sybil and file pollution attacks to describe the process of applying our model in hybrid P2P networks.

5.1. The detection of collusion and sybil attacks

By referring to Maze [27–29], we adopt the incentive mechanism that we increase a peer's score when the peer uploaded a file and decrease a peer's score when the peer downloaded a file to detect the peers which started collusion or Sybil attacks. Collusion peers increase their scores by mutually requesting files among themselves, and a Sybil node increases its score by using multiple accounts. According to [1], the set of keys can be set to I={duplication degree, pair-wise degree, PM ratio, traffic concentration degree}, as described in Section 3.2

According to the different features owned by normal peers, collusion peers and Sybil peers, we set the keys for different peers as follows. A normal peer's *BP* can be represented by I_1={low duplication degree, low pair-wise degree, low PM ratio, low traffic concentration degree}; a collusion peer's *BP* can be represented by I_2={high duplication degree, high pair-wise degree, low PM ratio, high traffic concentration degree}; a Sybil peer's *BP* can be represented by I_3={high duplication degree, high pair-wise degree, high PM ratio, high traffic concentration degree}. Note that here the words of "high" and "low" refer to which interval (see the paragraph of "Normalization of the initial behavior patterns" in Section 4.1) the value is in.

In some hybrid P2P networks, peers are clustered to form a subnet based on nodes' IPs aiming at reducing traffic overhead. The existing research [1] indicated that the nodes in the same IP space tend to build up each other, which means that a subnet might be abnormal in hybrid P2P networks.

Based on the above analysis, our strategy uses following steps to detect collusion and Sybil nodes

Step 1. For each subnet, its super node *SN* constructs the behavior patterns of the peers in the subnet and normalizes them to obtain peers' behavior pattern set, $D_{SN}^{T_x}$, within period T_x as mentioned in Section 4.1;

Step 2. We calculate the conditions used to determine whether the local *FP* mining process should be triggered or not, as described in Section 4.2.1.

Step 3. If the local *FP* mining process is triggered, the super node runs *Algorithm 1* mentioned in Section 4.2.1, to achieve top-*k* local *FPs*. Based on the mining results, we could reach a conclusion as follows.

(1) For a normal subnet, most nodes in the subnet are normal ones, and thus the local *FPs* would be the subset of I_1.
(2) For an abnormal subnet, most nodes in the subnet are abnormal ones, and thus the local *FPs* would be the subsets of I_2 and I_3.

Step 4. Each super node incrementally propagates its Local *FPs* among the super nodes and updates its own GlobalFPs based on the received and aggregated local *FPs* by using Algorithm 2 mentioned in Section 4.2.2.

Step 5. Each super node calculates the local outlier factor and the global outlier factor for the peers in the subnet the super node is in, and based on which to evaluate whether a peer is outlier or not as mentioned in Section 4.3 and to further distinguish the type of malicious peers as *Step 6* shows.

Step 6. Each super node analyzes the type of malicious nodes as follows.

(1) If the *BPs* of a malicious node conform to I_2, then the node is identified as the collusion node. Similarly, if the *BPs* of a malicious node conform to I_3, then the node is identified as the Sybil node;
(2) If the local *FPs* of an abnormal subnet conform to I_2, then there exists a collusion group in the subnet. Similarly, if the local *FPs* of an abnormal subnet conform to I_3, then there exist a lot of Sybil peers, each of which owns multiple accounts in the subnet.
(3) If the *BPs* of a malicious node do not conform to I_2 and I_3, then the node might be a malicious node of another type. In this situation, we should further analyze the node's *BPs*. For example, if a node's *BPs* conform to I={high duplication degree, low pair-wise degree, low PM ratio, low traffic concentration degree}, then the node might be a file polluter or an enthusiastic uploader of files. We could further analyze the node's behavior by taking the node's file upload frequency, repetitive interaction ratio and feedbacks into account. Details can be found in the next section.

5.2. The detection of file polluters

In order to detect file polluter, it is necessary to add several keys to the detection metrics. Thus, we set I={duplication degree, pair-wise degree, PM ratio, traffic concentration degree, file upload frequency, repetitive interaction ratio, feedback}, where *file upload*

Algorithm 1
Local frequent behavior pattern mining algorithm.

1) Calculate $\overline{D_{SN}^{\tau}}$ according to Formula (3);
2) Calculate $V_{SN}(\tau, \tau')$ and $MaxV_{SN}(\tau, \tau')$ according to Formulae (4)-(6);
3) If $(V_{SN}(\tau,\tau') \geq \mu_1 \text{ or } MaxV_{SN}(\tau,\tau') \geq \mu_2)$
   ① Execute the frequent behavior pattern mining algorithm to obtain top-k frequent behavior patterns from $\overline{D_{SN}^{\tau}}$;
      Input: k and $\overline{D_{SN}^{\tau}}$
      Output: the Top-k most frequent behavior patterns
   ② Set Update.inc and Update.del according to the newly mining result.
4) End

Algorithm 2
The update of GlobalFPs.

For each FP in Q_{SN}(Update.inc) and Q_{SN}(Update.del) do {

(1) When the *FP* in Q_{SN}(Update.inc) is new for GlobalFPs, we insert the *FP* into GlobalFPs as a new *FP*, and eliminate the *FP* from Q_{SN}(Update.inc);
(2) When the *FP* in Q_{SN}(Update.inc) is not new for GlobalFPs, we add Q_{SN}(Update.inc).FP-> IF to the field of GlobalFPs.FP->IF, and eliminate the *FP* from Q_{SN}(Update.inc);
(3) When the *FP* in Q_{SN}(Update.del) is new for GlobalFPs, we still keep it in Q_{SN}(Update.del), which will be used to update GlobalFPs next time. This case might be occurred due to the distributed message propagation and aggregation;
(4) When the *FP* in Q_{SN}(Update.del) is not new for GlobalFPs, we take the following three cases into account.

First, if (GlobalFPs.FP->IF - Q_{SN}(Update.del).FP->IF)>0, then we change GlobalFPs.FP->IF to the value of (GlobalFPs.FP->IF- Q_{SN}(Update.del).FP->IF), and eliminate the *FP* from Q_{SN}(Update.del);
Second, if (GlobalFPs.FP->IF - Q_{SN}(Update.del).FP->IF)<0, then we eliminate GlobalFPs.FP from GlobalFPs, and still keep Q_{SN}(Update.del).FP in Q_{SN}(Update.del) but change its *IF* to the value of (Q_{SN}(Update.del).FP->IF - GlobalFPs.FP->IF), which will be used to update GlobalFPs next time;
Third, if (GlobalFPs.FP->IF- Q_{SN}(Update.del).FP->IF)=0, then we eliminate the *FP* from the both sides;
}

frequency represents the frequency that a file is repeatedly uploaded, *repetitive interaction ratio* stands for the possibility that two nodes make transaction again, and *feedback* is the feedback on the received service.

Based on the above definition, we analyze the following two situations.

(1) From the angle of a file polluter

According to the above defined keys, a file polluter should have the following characteristics:

1) The file polluter repeatedly uploads polluted files with a higher frequency;
2) The receivers of the polluted files are distributed loosely over the network, so as to ease the spreading of the polluted files.

(2) From the angle of a polluted file receiver

The receiver of polluted files has the following characteristics:

1) For a normal receiver, the possibility that it repeatedly makes transactions with a polluter is lower.
2) For a normal receiver, it would give a file polluter a lower feedback.

According to the above analysis, a file polluter's behavior pattern can be represented by I_4={high duplication degree, low pair-wise degree, low PM ratio, low traffic concentration degree, high file upload frequency, low repetitive interaction ratio, low feedback}. This behavior pattern can be used to detect file polluters by using the similar approach as mentioned in Section 5.1.

6. Simulations and analysis

In this section, we evaluate our model's performance in a hybrid P2P network. We take a file sharing network as the application scenario of our model, and we construct the network similar to Maze [27–29] and KaZaA [21], which consists of super peers and ordinary peers. A peer joins the network by randomly selecting a super peer and uploads the meta-data of its shared files, including filename, file size, hash value of the file content, file descriptor and so on, to the super peer [24,25]. Each super peer holds its neighboring super peers' information. By referring to Maze [27–29], we adopt the incentive mechanism that we increase a peer's score when the peer uploaded a file and decrease a peer's score when the peer downloaded a file. We assume there are three types of attacks in the network, collusion, Sybil and file polluter, and all the malicious peers take the ratio of α. Under the incentive mechanism, the collusion peers and Sybil peers would behave maliciously to increase their scores. Specifically, the collusion peers increase their scores by mutually requesting files among themselves, and a Sybil node increases its score by using multiple accounts.

According to [1], we set I = {duplication degree, pair-wise degree, PM ratio, traffic concentration degree} as mentioned in Section 5.1. We set $h = 4$, meaning that each key could take one of the four values respectively representing very low, low, high and very high. There are 1000 nodes in the network, ten of which are randomly selected as the super peers, and the others are ordinary peers. Each simulation consists of 20 time windows, and each time window includes 5 periods.

For an ordinary peer, it completes the following tasks in a period. It sends a file request to its super peer, and waits for the service peers' list from the super peer; it downloads the requested file from a selected service peer, and gives a feedback on the downloaded file's quality to its super peer.

For a super peer, it completes the following tasks in a period. In the first period of the current time window, the super peer checks whether or not to run the local frequent pattern mining process according to the triggering conditions, and propagates the messages of Update.inc and Update.del to its neighboring super peers; processes the file requests coming from the ordinary peers; updates GlobalFPs according to the received messages coming from other super peers; establishes each ordinary peer's behavior patterns and calculates each ordinary peer's local outlier factor and global outlier factor, and based on which to detect and analyze malicious peers.

We use Peersim [30], an open source P2P systems simulator, to conduct the simulations. Each simulation runs 20 times, and the average value is reported as the simulation result. Without the loss of generality, we take the commonly used false positive rate (FPR, i.e. the ratio of peers that are normal but considered as malicious to all the normal peers) and false negative rate (FNR, i.e. the ratio of peers that are malicious but considered as normal to all the ma-

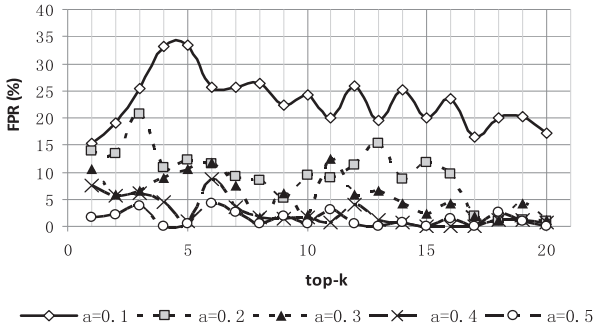

Fig. 1. Impacts of different top-k on FPR under different α.

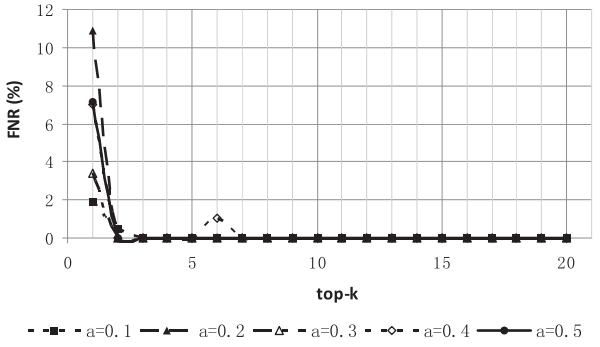

Fig. 2. Impacts of different top-k on FNR under different α.

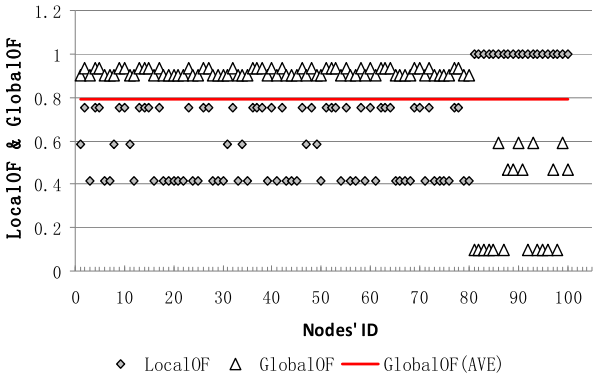

Fig. 3. Peers' LocalOF and GlobalOF in the abnormal subnet.

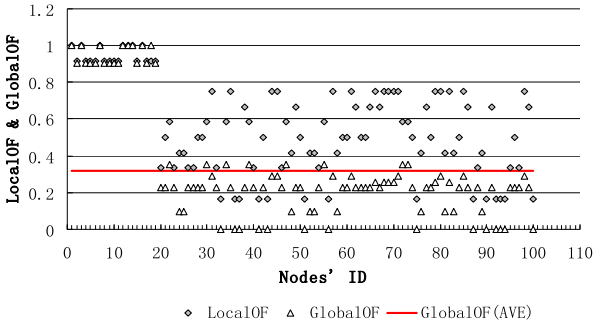

Fig. 4. Peers' LocalOF and GlobalOF in the normal subnet.

licious peers) as the criterion [11,12] to assess the performance of our model.

6.1. Impacts of different top-k on FPR and FNR under different α

In our model, top-k is a variable used to determine the size of LocalFP$_s$ mined in each subnet. In this section, we analyze the impacts of different top-k on FPR and FNR under different α, as shown in Figs. 1 and 2.

From Figs. 1 and 2 we see that different values of top-k have insignificant impacts on FPR and FNR. When the value of top-k is smaller, a part of normal FPs are excluded from the LocalFP$_s$ and the GlobalFP$_s$, which makes FPR and FNR higher. However, with the increase of top-k, GlobalFP$_s$ could fully reflect the behavior patterns of normal peers, which makes FPR and FNR drop and tend to be steady. As shown in Fig. 1, when $\alpha \leq 0.2$ and top-$k \geq 17$, FPR tends to be smaller and steady; when $\alpha \geq 0.3$ and top-$k \geq 14$, FPR also tends to be smaller and steady. As shown in Fig. 2, when top-$k \geq 2$, the value of FNR tends to 0, indicating that almost all the malicious nodes are detected even though the malicious node rate α is as much as 0.5.

Fig. 1 tells us that FPR becomes higher when α is smaller, and FPR becomes lower when α is bigger. This is because when α is smaller, the average outlier factor would be approximate to those of the normal peers and thus the outlier factors of some normal peers would be higher than the average outlier factor, which makes the normal peers whose outlier factors are equal to or higher than the average outlier factor mistakenly identified as the malicious peers, and so the FPR becomes higher. When α is bigger, the average outlier factor would be much higher than those of the normal peers, which makes the FPR lower.

6.2. Validation of peers' local outlier factors and global outlier factors

In the simulations, we only take two subnets into account, one is the abnormal subnet and another is the normal subnet. In the abnormal subnet, 80% of the nodes are assumed to be malicious and only 20% of the nodes are normal. In the normal subnet, 80% of the nodes are assumed to be normal and 20% of the nodes are malicious. The two subnets both consist of 100 nodes, and the other subnets are randomly formed. We set $\alpha = 0.2$ and top-$k = 3$. Fig. 3 and Fig. 4 show the simulation results.

In Fig. 3 and Fig. 4, the x-axis represents the node IDs, and the y-axis plots the average local outlier factor and the average global outlier factor of peers in the corresponding subnet. From Fig. 3 we see that the GlobalOF of most peers in the abnormal subnet is higher than the average value represented by GlobalOF(AVE), showing that the subnet is abnormal. In this situation, the malicious peers' LocalOF is lower and the normal peers' LocalOF is higher. This is because in such subnet the malicious peers' behavior patterns take the major part, which constructs the frequent behavior patterns. Consequently, the normal peers' LocalOF is higher and the abnormal peers' LocalOF is lower. In Fig. 4, the GlobalOF of most peers in the normal subnet is lower than the average value represented by GlobalOF(AVE), indicating that the subnet is normal. In this situation, malicious peers' LocalOF is higher and normal peers' LocalOF is lower. In the both situations, the peers whose GlobalOF is higher than the average value are the malicious peers and the peers whose GlobalOF is lower than the average value are the normal peers. The simulation results indicate that our model could accurately detect the normal subnet and the abnormal subnet, as well as the malicious peers in the two kinds of subnet.

6.3. Evaluation on the communication cost

In this simulation, we evaluate the communication cost generated by propagating Update.inc and Update.del among the super peers. As shown in Table 1, the message of Update.inc or Update.del consists of (FP_i, IF), so we use the average number of (FP_i, IF) propagated by each super node to quantify the communication cost. In Fig. 5, the x-axis plots the time windows, and the y-axis

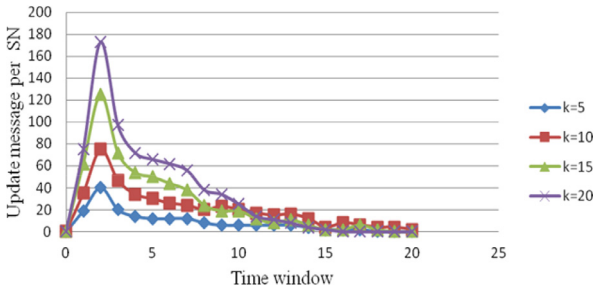

Fig. 5. Evolution of the communication cost under different Top-k.

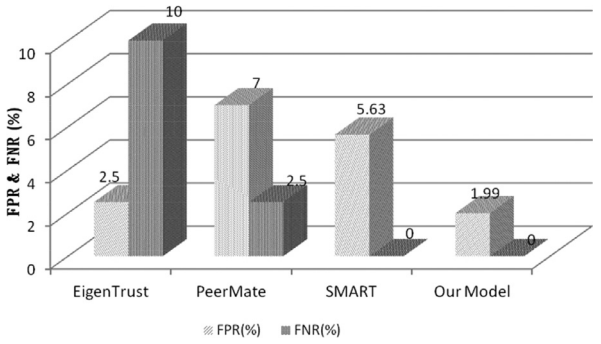

Fig. 6. FPR and FNR under different models with α=0.2.

represents the average number of (FP_i, IF) propagated by each super node. As shown in Fig. 5, in the initial phase of the simulation, the communication cost is correlated with the value of Top-k, and more messages are propagated among super peers due to the fact that in this phase each super peer needs to send almost all the FPs mined in its subnet to its neighboring super peers. However, as the simulation goes on, the propagated messages of each super peer reduce sharply and tend to zero. This is because the FPs stored in each super peer tend to be the same with the increase of time window. Note that a larger Top-k means more update messages should be propagated in the initial phase of the simulation, but meanwhile the propagated messages would reduce more sharply as the simulation goes on, since a larger Top-k makes the computation of GlobalFP more accurate, which means the GlobalFP stored in each super peer tends to be the same more quickly, and thus the propagated messages would reduce more sharply as the simulation goes on, as shown in Fig. 5.

6.4. Performance comparison

In this section, we examine the effectiveness of our model in comparison with EigenTrust [2], PeerMate [11] and SMART [12]. EigenTrust is a typical global trust model, and PeerMate and SMART are both trust-based malicious peer detection models, as mentioned in Section 2.

In this simulation, we set top-k=17 according to the simulation result given in Section 6.1. The value of α is respectively set to 0.2 and 0.4. We have all the malicious peers join the subnets randomly.

As shown in Fig. 6 and Fig. 7, the FNR of EigenTrust is the highest among the four models. EigenTrust iteratively calculates a peer's trust by using the collected feedbacks. Due to the sparseness of the subjective feedbacks, the correctness of its trust calculation could not be guaranteed. As mentioned in Section 2, PeerMate could not identify a part of Sybil nodes which have the similar behavior to the normal peers. This makes the FNR of PeerMate higher than that of SMART. Note that though the FNR of PeerMate and SMART is smaller than that of EigenTrust, their FPR is the

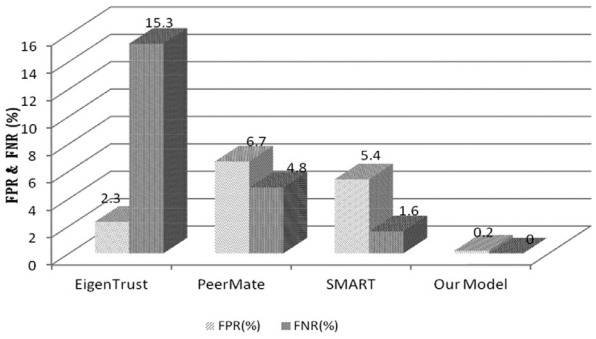

Fig. 7. FPR and FNR under different models with α=0.4.

highest among the four models. This is because the two models mistakenly identified the low-ranking normal nodes as the malicious ones. Our model has the lowest FNR and FPR among the four models, mainly because we do not use trust model but take the actual transaction data as the outlier mining basis to detect malicious peers, which could not be affected by the peers' subjective behavior such as feedbacks. From Figs. 6 and 7, we see that the higher value of α makes the FPR of our model reduced significantly. This is because when the number of malicious nodes is higher, the average global outlier factor tends to be the value of the average malicious peers' outlier factor which is significantly greater than that of the normal peers, and thus the value of FPR becomes smaller.

7. Conclusions

In this paper, we mainly discussed how to detect malicious peers using outlier mining approach in hybrid P2P networks. We first presented several definitions, and described a peer's behavior patterns based on the peer's interaction data. Then, we detailed the local frequent behavior pattern mining process and the global frequent behavior pattern producing approach by incrementally propagating and aggregating the local frequent behavior patterns. Based on the local frequent patterns and the global frequent patterns, we depicted the malicious node detection process and the examples of using our model. The simulation results indicated that our model could effectively detect malicious behavior, such as collusion, Sybil and file polluter. In our future work, we will focus our efforts on both the settings of keys used to perform frequent behavior pattern mining and the application of our model in other types of P2P networks.

References

[1] Q. Lian, Z. Zhang, M. Yang, et al., An empirical study of collusion behavior in the maze P2P file-sharing system, in: ICDCS 27th International Conference on Distributed Computing Systems, Toronto, Canada, IEEE, 2007, pp. 1–10.
[2] S.D. Kamvar, M.T. Schlosser, H.G. Molina, The eigentrust algorithm for reputation management in P2P networks, in: Proc. of the12th Int'1 World Wide Web Conf., Hungary, ACM Press, 2003, pp. 2–4.
[3] A. Das, M.M. Islam, SecuredTrust: a dynamic trust computation model for secured communication in multiagent systems, IEEE Trans. Depend. Secure Comput. 9 (2) (2012) 5.
[4] X.Y. Li, F. Zhou, X.D. Yang, Scalable feedback aggregating (SFA) overlay for large-scale P2P trust management, IEEE Trans. Parallel Distrib. Syst. 23 (10) (2012) 7–10.
[5] W. Dou, H. Wang, Y. Jia, P. Zou, A recommendation-based Peer-to-Peer trust model, J. Softw. 15 (4) (2004) 571–583.
[6] Z. Li, H.Y. Shen, K. Sapra, Leveraging social networks to combat collusion in reputation systems for Peer-to-Peer Networks, IEEE Trans. Comput. 62 (9) (2013) 1745–1758.
[7] B. Viswanath, K.P. Gummadi, An analysis of social network-based Sybil defenses, in: ACM SIGCOMM Computer Communication Review - SIGCOMM '10, 40, 2010, pp. 363–374.
[8] X.F. Meng, Y.L. Ding, Y. Gong, @Trust: a trust model based on feedback-arbitration in structured P2P Network, Comput. Commun. 35 (16) (2012) 2044–2053.
[9] L. Mekouar, Y. Iraqi, R. Boutaba, Peer-to-Peer's most wanted: malicious peers, Comput. Netw. 50 (4) (2005) 545–562.

[10] G.M. Cai, M. Wang, Y.D. Wang, et al., A collusion detection trust model based on behaviour similarity, in: Information and Communications Technologies (IETICT 2013), IET International Conference on, Beijing, IEEE, 2013, pp. 241–245.
[11] X.L. Wei, T. Ahmed, M. Chen, et al., PeerMate: a malicious peer detection algorithm for P2P systems based on MSPCA, in: ICNC International Conference on Computing, Networking and Communications, HI, IEEE Commun, 2012, pp. 815–819.
[12] X.L. Wei, J.H. Fan, M. Chen, et al., SMART: a subspace based malicious peers detection algorithm for P2P systems, Int. J. Commun. Netw. Inf. Secur. 5 (1) (2013) 1–8.
[13] Y.K. Li, J.C.S. Lui, On detecting malicious behaviors in interactive networks: algorithms and analysis, in: COMSNETS 2012 2012 Fourth International Conference on Communication Systems and Networks, IEEE Commun, Bangalore, 2012, pp. 1–10.
[14] A. Xue, L. Yao, S. Ju, et al., Survey of outlier mining, Comput. Sci. 35 (1) (2008) 13–18.
[15] Z. He, X. Xu, J.Z. Huang, et al., FP-Outlier: frequent pattern based outlier detection, Comput. Sci. Inf. Syst. 2 (1) (2005) 103–118.
[16] X.Y. Zhou, Z.H. Sun, B.L. Zhang, et al., A fast outlier detection algorithm for high dimensional categorical data streams, J. Softw. 18 (4) (2007) 933–942.
[17] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: SIGMOD'93 Proceedings of the 1993 ACM-SIGMOD international conference on management of data, Washington, DC, ACM, 1993, pp. 207–216.
[18] Agrawal R., Srikant R. Fast algorithms for mining association rules. VLDB'94 Proceedings of the 1994 international conference on very large data bases. Santiago: VLDB, 1994:487–499.
[19] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, in: SIGMOD'00 Proceeding of the 2000 ACM-SIGMOD international conference on management of data, Dallas, ACM, 2000, pp. 1–12.
[20] A. Salam, M.S.H. Khayal, Mining top-k frequent patterns without minimum support threshold, Knowl. Inf. Syst. 30 (1) (2012) 57–86.
[21] J. Liang, R. Kumar, K.W. Ross, The fast-track overlay: a measurement study, Comput. Netw. 50 (6) (2006) 842–858.
[22] K. Tutschku, A measurement-based traffic profile of the eDonkey file sharing service, in: 5th International Workshop, Springer, France, 2004, pp. 12–21.
[23] D. Huang, A. Zhang, J. Huang, J. Li, Research on Peer-to-peer network Gnutella 0.6 architecture, Comput. Applicat. Softw. 25 (6) (2008) 208–210.
[24] Y. Jin, Y. Liu, H.W. Zhao, Trust-Based supernode selection in Peer-to-Peer Systems, ICFCC Future Computer and Communication 2010 2nd International Conference on, IEEE, Wuhan, 2010 V1–285-V1-289.
[25] H.F. Luo, L. Deng, Research on a P2P supper node selection mechanism based on trust model, in: ICCSE The 8th International Conference on Computer Science & Education, Colombo, IEEE, 2013, pp. 851–854.
[26] J. Han, D. Park, A lightweight personal grid using a supernode network, in: P2P'03 Proceedings of the Third International Conference on Peer-to-Peer Computing, IEEE, 2003, pp. 168–175.
[27] M. Yang, Z. Zhang, X. Li, Y. Dai, An empirical study of free-riding behavior in the maze P2P file-sharing system, in: Proceedings of IPTPS, Ithaca, NY, February 2005.
[28] H. Chen, M. Yang, et al., Maze: a social Peer-to-peer network, The International Conference on e-Commerce Technology for Dynamic e-Business (CEC-EAST'04), September 2004.
[29] Maze [EB/OL]. <http://maze.tianwang.com/>.
[30] Peersim [EB/OL]. <http://peersim.sourceforge.net/>.

Xianfu Meng received his B.E. and M.E. degrees in Computer Science and Technology from Dalian University of Technology, China in 1983 and 1986, respectively. His current research interests include the peer-to-peer computing and the distributed systems.

Shuang Ren is a graduated student and received the M.E. degree from Dalian University of Technology, China in 2015. Her research interests focus on anti-attacks in P2P networks.