Detecting Malicious Peers in A Reputation-Based Peer-to-Peer System

Loubna Mekouar, Youssef Iraqi and Raouf Boutaba
University of Waterloo, Waterloo, Canada
{lmekouar, iraqi, rboutaba}@bbcr.uwaterloo.ca

***Abstract*— In this paper we propose a reputation management scheme for partially decentralized peer-to-peer systems. The reputation scheme helps building trust between peers based on their past experiences and feedbacks from other peers. Our system is novel in that it is able to detect not only malicious peers sending inauthentic files but also malicious peers that are lying in their feedbacks. To detect those peers, we introduce the new concept of *suspicious transactions*. The simulation results show that the proposed scheme is able to effectively detect malicious peers and isolate them from the system, hence reducing the amount of inauthentic uploads and increasing peers' satisfaction.**

I. Introduction

Several reputation management systems have been proposed in the literature [1], [2], [3], [4], [5]. All of these have focused on the completely-decentralized P2P systems. No reputation management system has been proposed for partially-decentralized P2P systems. Only KaZaA, a proprietary partially-decentralized P2P system, has introduced basic reputation metric (called "participation level") for rating peers. The proposed reputation management schemes for completely-decentralized P2P systems cannot be applied in the case of a partially-decentralized systems. Partially-decentralized P2P systems (e.g. KaZaA [6], Morpheus [7] and Gnutella2 [8]), have been proposed to reduce the control overhead needed to run the P2P system. In these systems, some of the peers, that are called "supernodes" or "ultrapeers", index the files shared by peers connected to them, and proxy search requests on behalf of these peers [9]. Queries are therefore sent to supernodes, not to other peers. A supernode typically supports 300 to 500 peers depending on available resources [8].

In [10], we proposed a reputation management system for partially-decentralized P2P systems. This reputation mechanism allows a more clearsighted management of peers and files. Good reputation is obtained by having consistent good behavior through several transactions. The reputation criterion is used to distinguish between peers. The goal is to maximize the user satisfaction and decrease the sharing of corrupted files. In the following, we refer to the Real Behavior Based Algorithm in [10] as the Inauthentic Detector Algorithm (*IDA*). This algorithm detects malicious peers who are sending inauthentic files and isolates them from the system.

In all the previously proposed feedback-based reputation management schemes for P2P systems, the emphasize was on detecting and punishing peers who are sending inauthentic files. No special mechanism was proposed to detect and punish peers that send wrong feedbacks.

Indeed, peers can lie in their feedbacks. Although some proposed feedback-based reputation schemes take this behavior into consideration, those schemes rely on peers' reputation for their peer-selection process.

Such liar peers can subvert the reputation system by affecting badly the reputation of other peers (increase the reputation of malicious peers, or decrease the reputation of good peers). These malicious peers may not be detected if they are not sending inauthentic files and, hence, their reputation can be high and they will be trusted by the system.

We believe that it is of paramount importance to detect liar peers and prevent them from affecting the system.

In this paper, we propose a new scheme called the Malicious Detector Algorithm (*MDA*), that in addition to detecting and punishing inauthentic peers (based on *IDA*), detects liar peers and punishes them. The new scheme reduces considerably the amount of malicious uploads and protects the health of the system.

The paper is organized as follows. In Section II, we introduce the reputation management scheme considered in this paper. Section III, presents an analysis of peers' behavior. Section IV discusses the proposed approaches to detect malicious peers, while Section V presents the performance evaluation of the proposed scheme. Section VI presents the related works and finally, Section VII concludes the paper.

II. Reputation Management

A. Notations and Assumptions

- Let P_i denotes peer i
- Let $D_{i,j}$ be the units of downloads performed from peer P_j by peer P_i
- Let $D_{i,*}$ denotes the units of downloads performed by peer P_i
- Let $D_{*,j}$ denotes the units of downloads from peer P_j, i.e. the units of uploads by peer P_j
- Let $A^F_{i,j}$ be the appreciation of peer P_i for downloading the file F from peer P_j.
- Let $Sup(i)$ denotes the supernode of peer i

B. The Reputation Management Scheme

In [10], we have proposed a reputation management scheme that is based on the following mechanism. After downloading a file F from peer P_j, peer P_i will value this download. If the

Type	Peer	Inauthentic Behavior	Liar Behavior
$T1$	Good	Low	Low
$T2$	Malicious: Inauthentic Peer	High	Low
$T3$	Malicious: Liar Peer	Low	High
$T4$	Malicious: Inauthentic and Liar Peer	High	High

TABLE I

PEER BEHAVIOR

file received corresponds to the requested file and has good quality, then we set $A^F_{i,j} = 1$. If not, we set $A^F_{i,j} = -1$. In this case, either the file has the same title as the requested file but different content, or that its quality is not acceptable.

Each peer P_i in the system has, at least, two values, called *reputation data* (REP_{P_i}), stored by its supernode $Sup(i)$:

1) $D^+_{*,i}$: Successful uploads from P_i to other peers,
2) $D^-_{*,i}$: Failed uploads from peer P_i to other peers

Once the peer sends its appreciation, the size of the download $Size(F)$ (the amount of bytes downloaded by the peer P_i from the peer P_j) is also sent[1]. The reputation data of P_j is updated based on the amount of data downloaded. In this case, after each download transaction by peer P_i from peer P_j, $Sup(j)$ will perform the following operation after receiving the appreciation from $Sup(i)$:

$$\begin{array}{ll} \text{If } A^F_{i,j} = 1 & \text{then } D^+_{*,j} = D^+_{*,j} + Size(F) \\ & \text{else } D^-_{*,j} = D^-_{*,j} + Size(F) \end{array} \quad (1)$$

In this scheme, we compute the reputation (called Authentic Behavior) of a peer P_j as:

$$\begin{array}{ll} AB_j = \frac{D^+_{*,j} - D^-_{*,j}}{D^+_{*,j} + D^-_{*,j}} = \frac{D^+_{*,j} - D^-_{*,j}}{D_{*,j}} & \text{if } D_{*,j} \neq 0 \\ AB_j = 0 & \text{otherwise} \end{array} \quad (2)$$

Note that the reputation as defined in equation 2 is a real number between -1 (if $D^+_{*,j} = 0$) and 1 (if $D^-_{*,j} = 0$).

When a peer P_i joins the system for the first time, all values of its *reputation data* REP_{P_i} are initialized to zero.

The following is the life cycle of a peer P_i in our reputation-based P2P system:

1) Send a request for a file F to the supernode $Sup(i)$
2) Receive a list of candidate peers that have the file F
3) Select a peer or a set of peers P_j based on a reputation metric (see equation 2)
4) Download the file F
5) Send the feedback $A^F_{i,j}$. $Sup(j)$ will update the reputation data REP_{P_j}.

III. PEER BEHAVIOR UNDERSTANDING

A. Peer Behavior Categorization

In a P2P system, we consider two general behaviors of peers: good and malicious. Good peers are those that send authentic files and do not lie in their feedbacks (Type $T1$ in Table I). Malicious peers can be divided into three categories: 1) peers that send inauthentic files and do not lie in their feedbacks (Type $T2$), 2) peers that send authentic files and do lie in their feedbacks (Type $T3$), and 3) peers that send inauthentic files and do lie in their feedbacks (Type $T4$).

A liar peer is one that after receiving an authentic file, instead of giving an appreciation equal to 1, the peer sends an appreciation equal to -1 to decrease the reputation of the peer uploading the file. Or, if the peer receives an inauthentic file, it sends a positive appreciation to increase the reputation of other malicious peers.

Note that we consider the consistent behaviors of peers. This means that most of the time a peer behavior is consistent with the category it belongs to (i.e. $T1$, $T2$, $T3$, or $T4$). For example, a good peer can sometimes (on purpose or by mistake) send inauthentic files. Note also that peers can change their behavior over time and hence can jump from one category to another.

Free riders [11] can also be considered as malicious peers. In this paper, we do not consider free riders as malicious if they do not affect directly the reputation of other peers. A free rider can belong to one of the categories described in Table I and the system will deal with it accordingly.

B. Effect On Reputation

Peers can have positive or negative reputations. A good peer usually has a positive reputation since he is behaving well, but since malicious peers can lie, his reputation can decrease and even get negative. In contrast, malicious peers will have negative reputations since they are sending inauthentic files. However, their reputation can increase and even get positive if some other malicious peers send positive appreciations even if they receive inauthentic files. This happens in systems where the liar peers are not detected nor punished.

IV. DETECTING MALICIOUS PEERS

Let's assume that a peer P_i downloads a file F from a peer P_j. We focus on the authentic behavior (sending authentic or inauthentic files) of peer P_j since he is sending the file, and the credibility behavior (lying or not in the feedback) of peer P_i since he is sending the appreciation that will affect the reputation of peer P_j. If we want to take the appropriate actions after this transaction, we have to detect if peer P_j belongs to any of the categories $T2$ and $T4$, and if peer P_i belongs to any of the categories $T3$ and $T4$.

IDA [10] allows us to detect peers sending inauthentic files. The goal now is to detect peers that send wrong feedbacks and diminish their impact on the reputation-based system.

A. First Approach

One approach is to say that malicious peers have a low reputation than good peers. One way of reducing the impact of peers having a low reputation is to take this later into account when updating the reputation of other peers.

[1] Alternatively the supernode can know the size of the file from the information received as a response to the peer's request.

We can then change operation (1) to:

$$\begin{array}{ll}\text{If } A^F_{i,j} = 1 & \text{then } D^+_{*,j} = D^+_{*,j} + \frac{1+AB_i}{2} \times Size(F) \\ & \text{else } D^-_{*,j} = D^-_{*,j} + \frac{1+AB_i}{2} \times Size(F)\end{array} \quad (3)$$

Using this approach[2], the impact of peer P_i on the reputation of peer P_j is related to the trust given to peer P_i. The trust is expressed by the value of AB_i. If peer P_i has a good reputation (usually above zero), he is trusted more and he will impact the reputation of peer P_j, but, if his reputation is low (usually negative), only a small fraction of the file size is considered hence reducing the impact on the reputation of peer P_j.

In case peer P_i is new, his reputation is null and since we do not know yet if he is a good or a malicious peer, only half of the size of the uploaded file F is affecting the reputation of the peer uploading the file (i.e. peer P_j).

The problem with this approach appears in the following example. Assume that some peers belong to category $T3$. Those peers always send authentic files, but send also wrong appreciations. Most of the time, and according to operation (3), those peers will have a high reputation since they always send authentic files and hence will receive good feedbacks[3]. Those peers will be trusted by the system and will affect badly the reputations of other peers and may eventually brake the system. The performance evaluation section assesses the effect of liar peers on the reputation of other peers.

B. *Second Approach*

Another approach to detect the peers that lie in their feedbacks is to detect *suspicious transactions*. A *suspicious transaction* is one in which the appreciation is different from the one expected knowing the reputation of the sender. In other words, if $A^F_{i,j} = 1$ and $AB_j < 0$ or if $A^F_{i,j} = -1$ and $AB_j > 0$ then we consider this transaction as suspicious.

To detect peers that lie in their feedbacks, for each peer P_i we keep track of the following values:

1) N_i: The total number of downloads performed by peer P_i
2) N^*_i: The number of downloads by peer P_i where the sign of the appreciation sent by peer P_i is different from the sign of the sender's reputation, i.e.$A^F_{i,j} \times AB_j < 0$

Note that $N^*_i \leq N_i \ \forall i$

When receiving the appreciation (i.e. $A^F_{i,j}$) of peer P_i, its supernode $Sup(i)$ will update the values of N_i and N^*_i as follows:

$$\begin{array}{l} N_i = N_i + 1 \\ \text{If } (A^F_{i,j} \times AB_j) < 0 \text{ then } N^*_i = N^*_i + 1 \end{array} \quad (4)$$

Let α_i be the ratio of N^*_i and N_i:

$$\alpha_i = \frac{N^*_i}{N_i} \quad (5)$$

Note that $0 \leq \alpha_i \leq 1 \ \forall i$

α_i is the ratio of the number of suspicious feedbacks[4] sent by peer P_i over the total number of feedbacks sent by peer P_i. α_i is a good indicator of the liar behavior of peer P_i. Indeed, if peer P_i lies in its feedbacks, the number of times $A^F_{i,j}$ and the sender's reputation having different signs, is high and hence the value of N^*_i. Liar peers will tend to have values of α_i near 1. Good peers will tend to have values of α_i near zero.

To minimize the effect of liar peers, we propose to use the following update strategy for the sender's appreciation; After receiving the appreciation $A^F_{i,j}$, the sender's supernode $Sup(j)$ will perform the following operation:

$$\begin{array}{ll}\text{If } A^F_{i,j} = 1 & \text{then } D^+_{*,j} = D^+_{*,j} + (1-\alpha_i) \times Size(F) \\ & \text{else } D^-_{*,j} = D^-_{*,j} + (1-\alpha_i) \times Size(F)\end{array} \quad (6)$$

Since liar peers (in categories $T3$ and $T4$) will have a high value of α_i, their effect on the reputation of the peer sending the file is minimized. This is because the higher the value of α_i the lower the value of $(1 - \alpha_i)$. On the other hand, good peers will have a lower value of α_i and hence will keep having an impact of the reputation of other peers.

Note that AB_j is updated after each upload of peer P_j and α_i is updated after each download of peer P_i. This means that liar peers will be punished even if they did not upload any file and inauthentic peers will be punished even if they did not perform any download.

If a peer P_i changes its behavior, α_i will change also and hence its impact on the reputation of others. For example, if a peer P_i changes its behavior from category $T3$ to $T1$, the number of suspicious transactions N^*_i involving this peer (in comparison to the total number of transactions N_i) will decrease and hence the value of α_i will decrease also, making the impact of this peer more considerable.

Let the *Credibility Behavior* of peer P_i be: $CB_i = 1 - \alpha_i$.

In this case, the reputation of a peer P_i is the couple (AB_i, CB_i) which characterize the behavior of peer P_i in terms of *Authentic Behavior* (sending authentic or inauthentic files) and *Credibility Behavior* (lying or not in the feedback). Note that because the behavior of the peers is characterized by two values, the supernode can still download a file from a peer with low value of CB_i as long as the value of AB_i is high. This means that the system can still take advantage of a peer that provides authentic files but lies in its feedbacks. We will refer to this algorithm as the Malicious Detector Algorithm (*MDA*).

The performance evaluation section presents results that show that the *Credibility Behavior* is a very good indicator of the liar behavior of peers, and hence can be used to differentiate between peers. This will in turn allow for a better management of peers and hence provide better performance.

This new way of detecting malicious peers, will allow the supernode to enforce service differentiation according to peers' reputation. For example, when processing a search request, the supernode can give higher priority to good peers.

[2] In operation (3), $\frac{1+AB_i}{2}$ can be replaced by any function of AB_i that is strictly increasing from 0 to 1

[3] We assume that the percentage of malicious peers, in a P2P system, is lower than the percentage of good peers. This assumption is realistic since this is the basis on which peer-to-peer systems can work

[4] A suspicious feedback is the feedback sent during a suspicious transaction

Category	Percentage of peers	Probability of sending inauthentic files	Probability of sending wrong feedbacks
G	40%	1%	1%
$M1$	30%	50%	50%
$M2$	30%	90%	90%

TABLE II
PEER BEHAVIOR

V. PERFORMANCE EVALUATION

A. Simulated Algorithms

We will simulate the Malicious Detector Algorithm proposed in this paper. We will compare its performance with the *IDA* scheme [10] and with the following two schemes.

In KaZaA [6], the peer participation level is computed as follows: $(uploaded/downloaded) \times 100$, i.e. using our notation (cf. section II-A) the participation level is $(D_{*,j}/D_{j,*}) \times\ 100$. We will consider the scheme where each peer uses the participation level of other peers as a selection criterion and we will refer to it as the KaZaA-Based algorithm (*KB*).

We will also simulate a system without reputation management. This means that the selection is done in a random way. We will refer to this algorithm as the Random Way algorithm (*RW*).

B. Simulation Parameters

We use the following simulation parameters:

- We simulate a system with 1000 peers.
- The number of files is 1000.
- File sizes are uniformly distributed between 10MB and 150MB.
- At the beginning of the simulation, each peer has 30 randomly chosen files and each file has at least one owner.
- File requests follow the real life distribution observed in [12]. This means that each peer can ask for a file with a Zipf distribution over all the files that the peer does not already have. The Zipf distribution parameter is chosen close to 1 as assumed in [12].
- Peers behavior and distribution are as depicted in table II
- Only 40% of all peers with the requested file are found in each request.
- We simulate 30000 requests. This means that each peer performs an average of 30 requests. For this reason we do not specify a storage capacity limit for the peers.
- The simulations were repeated 10 times over which the results are averaged.

According to table II, peers with indices from 1 to 300 belong to category $M2$, peers with indices from 301 to 600 belong to category $M1$ and peers with indices from 601 to 1000 belong to category G.

We have considered a situation where we have a high percentage of malicious peers to show the effectiveness of our proposed scheme.

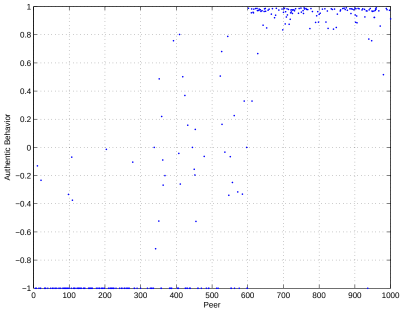

Fig. 1. Authentic Behavior with *IDA* (with no liar peers)

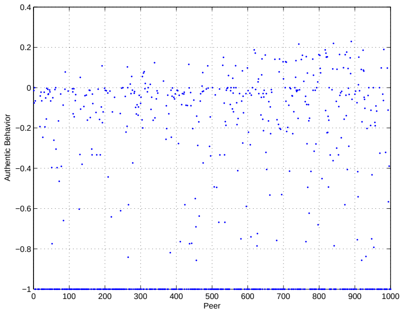

Fig. 2. Authentic Behavior with *IDA* (with liar peers)

C. Performance Parameters

In our simulations we will mainly focus on the following performance parameters:

1) The peer satisfaction: computed as the difference of non-malicious downloads and malicious ones over the sum of all the downloads performed by the peer. The peer satisfaction is averaged over all peers.
2) The percentage of malicious uploads: computed as the sum of the size of all malicious uploads performed by all peers during the simulation over the total size of all uploads.

D. Simulation Results

Figures 1 and 2 show the *Authentic Behavior* values for peers when using *IDA*. Figure 1 presents the results in a situation where no peer lies in its feedbacks, while figure 2 shows the results where there are liar peers in the system. The distribution of peers' behavior in the case where no liar peers exist is the same as in table II with the fourth column set to zero in all categories.

It is clear from figure 1 that *IDA* is able to differentiate between the peers and detect those that send inauthentic files. Good peers (with indices from 601 to 1000) have high AB values while malicious peers (from 1 to 600) have low AB values (most of peers with indices between 1 and 300 have a value of -1). However, if liar peers exist, those peers affect badly the system and makes it difficult to differentiate between peers (c.f. figure 2). This affects greatly the performance of the system as will be shown in figure 4.

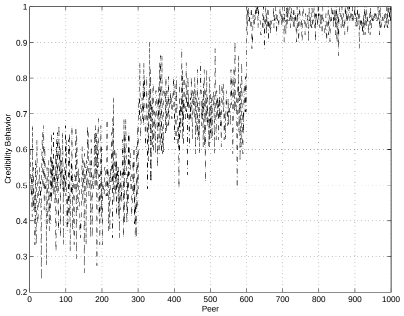

Fig. 3. Credibility Behavior

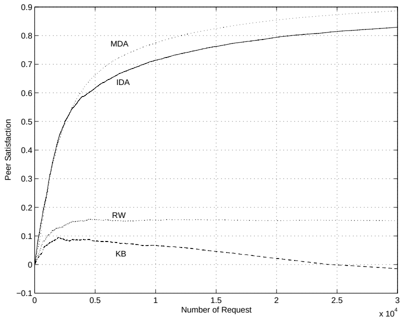

Fig. 4. Peer Satisfaction

Figure 3 depicts the *Credibility Behavior* of the peers when using *MDA*. The figure shows that CB is a very good indicator of the liar behavior of the peers. Indeed, good peers (with indices from 601 to 1000) have a very high value of credibility while liar peers (from 1 to 600) have lower values. This indicator is also able to differentiate between degrees of liar behavior; peers with lower probability of lying (indices from 301 to 600) have higher credibility than those with higher probability of lying (indices 1 to 300).

Figure 4 depicts the peer satisfaction achieved by the four considered schemes. The X axis represents the number of requests while the Y axis represents the peer satisfaction. Note that the maximum peer satisfaction that can be achieved is 1. Note also that the peer satisfaction can be negative. According to the figure, it is clear that the *MDA* and *IDA* schemes outperform the *RW* and *KB* schemes in terms of peer satisfaction. The bad performance of *KB* can be explained by the fact that it does not distinguish between malicious and non-malicious peers. As long as the peer has the highest participation level, it is chosen regardless of its behavior. The *RW* scheme chooses peers randomly and hence the results observed from the simulations (i.e. 15% satisfaction) can be explained as follows. With the values of table II, we can expect to have $(99\% \times 40\% + 50\% \times 30\% + 10\% \times 30\% =)$ 57.6% of authentic uploads and $(1\% \times 40\% + 50\% \times 30\% + 90\% \times 30\% =)$ 42.4% inauthentic uploads in average. As the peer satisfaction is computed as the difference of non-malicious downloads and malicious ones over the sum of all the downloads performed by the peer. We can expect a peer satisfaction of $(57.6 - 42.4)/(57.6 + 42.4) = 15.2\%$.

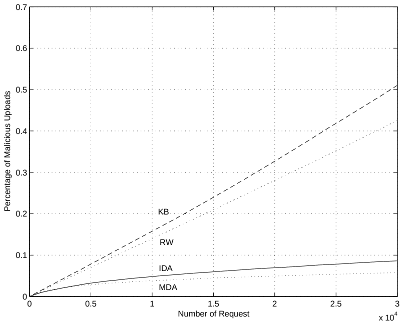

Fig. 5. Percentage of malicious uploads

Our schemes (*MDA* and *IDA*) make the distinction and do not chose a peer if it is detected as malicious. Since *MDA* is able to detect liar peers, it can protect the system from them and hence is able to take the right decision when choosing a peer to download from. In this new algorithm, the Authentic Behavior of a peer P_i (i.e. AB_i) reflects better the real behavior of this peer. In *IDA*, however, liar peers affect the Authentic Behavior values of other peers and hence lower the achieved peer satisfaction.

Figure 5 shows the percentage of malicious uploads, i.e. the percentage of inauthentic file uploads. As in *RW* scheme peers are chosen randomly, we can expect to see a steady increase of the percentage of malicious uploads. In *KB* scheme, the peer with the highest participation level is chosen. If the chosen peer happens to be malicious, the size of malicious uploads will increase dramatically as malicious peers are chosen again and again. This is reflected in figure 5 where *KB* has worse results than *RW*.

IDA can quickly detect inauthentic peers and avoid choosing them for uploads. This isolates the inauthentic peers and controls the size of malicious uploads. However, since *IDA* does not detect liar peers, the reputation of peers is affected as shown in figure 2. This will sometimes result in bad decisions. *MDA* on the other hand takes into consideration liar behavior and thanks to the *Credibility Behavior* parameter, is able to reduce the effect of liar peers on the system. This allows the system to take more clearsighted decisions. This, of course, results in using the network bandwidth more efficiently and higher peer satisfaction as shown in figure 4. Figure 6 shows that the new scheme achieves a 33.55% improvement in comparison to *IDA*. This gain will continue to increase with the number of requests as *MDA* makes more and more good decisions.

Note that our scheme achieves good performance even if we have a high number of malicious behaviors. As stated earlier, without any reputation management scheme we can expect 42.4% of inauthentic uploads. After the 30000 requests considered, our scheme reduces this to about 6% with a peer satisfaction of almost 90%.

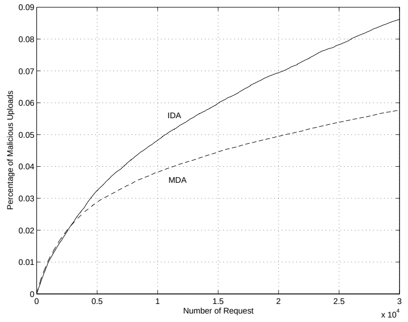

Fig. 6. Percentage of malicious uploads for MDA and IDA

VI. Related Works

eBay [13] uses the feedback profile for rating their members and establishing the members' reputation. Members rate their trading partners with a positive, negative or neutral feedback, and explain briefly why. The reputation is calculated by assigning 1 point for each positive comment, 0 points for each neutral comment and -1 point for each negative comment. In eBay, no special mechanism is provided to detect members that lie in their feedbacks.

In [2], the distributed polling algorithm P2PRep is used to allow a servant *p* looking for a resource to enquire about the reputation of offerers by polling its peers. In the basic polling, peers send their opinion and *p* uses the vote to determine the best offerer. In the enhanced polling, the peers provide their own opinion about the reputation of the offerers in addition to their identities. This later will be used by *p* to weight the vote received. This scheme incurs considerable overhead by polling peers for their votes. Moreover, each peer has to keep track of past experiences with all other peers. In addition, the reputation of the peers is used to weight their opinions, however, as we have shown earlier the reputation score is not enough to reflect the credibility of a peer.

In [3], the EigenTrust algorithm assigns to each peer in the system a global trust value. This value is based on its past uploads and reflects the experiences of all peers with the peer. This scheme is a reactive one, it requires reputations to be computed *on-demand* which requires cooperation from a large number of peers in performing computations. As this is performed for each peer having the requested file with the cooperation of all other peers, this will introduce additional latency as it requires long periods of time to collect statistics and compute a global rating. Most of the proposed reputation management schemes for completely decentralized P2P systems are reactive and hence suffer from this drawback. Moreover, they tend to consider the reputation as a score for the credibility of a peer which was shown to be ineffective.

In [14], the proposed algorithms use only limited information sharing between nodes. Each node records statistics and ratings regarding other peers. As the node receives and verifies files from peers, it updates the stored data. Nodes can share their opinions and incorporate them in their ratings. In the proposed voting reputation system, the querying node receives ratings from peers and weights them accordingly to the ratings that the peer has for these peers to compute a quorum rating. The peers can be selected from the neighbor list (Neighbor-voting) or from the friend list (Friend-voting). In the latter case, friends are chosen from peers who have proven to be reputable. Note that a peer can be reputable (high authentic behavior), but not credible. In [14], no mechanism is given to detect liar peers.

VII. Conclusion

In this paper, we propose a new reputation management scheme for partially decentralized P2P systems. Our scheme is based on mechanisms to detect peers that are sending inauthentic files and those that lie in their feedbacks. The new concept of *suspicious transactions* is introduced to detect liar peers. To our knowledge, we are the first to represent the reputation of peers using two values, one for the *Authentic Behavior* and one for the *Credibility Behavior*, which characterize more effectively the real behavior of peers. Performance evaluation shows that our scheme is able to detect and isolate malicious peers from the system hence providing higher satisfaction and better network bandwidth utilization. Our reputation management scheme is simple and proactive. Furthermore, it does not require any synchronization between the peers.

References

[1] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," in *The 9th International Conference on Information and Knowledge Management*, Atlanta, USA, November 2001, pp. 310–317.

[2] F. Cornelli, E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati, "Choosing Reputable Servents in a P2P Network," in *The 11th International World Wide Web Conference*, Honolulu, USA, May 2002, pp. 376–386.

[3] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust Algorithm for Reputation Management in P2P Networks," in *The 12th International World Wide Web Conference*, Budapest, Hungary, May 2003, pp. 640–651.

[4] M. Gupta, P. Judge, and M. Ammar, "A Reputation System for Peer-to-Peer Networks," in *ACM 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Monterey, USA, June 2003, pp. 144–152.

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

[6] "Kazaa," http://www.kazaa.com/.

[7] "Morphus," http://www.morphus.com/morphus.htm.

[8] "Gnutella2 Specification," http://www.gnutella2.com/.

[9] S. Androutsellis-Theotokis, "A Survey of Peer-to-Peer File Sharing Technologies," Tech. Rep., ELTRUN, 2002.

[10] L. Mekouar, Y. Iraqi, and R. Boutaba, "A Reputation Management and Selection Advisor Schemes for Peer-to-Peer Systems," in *The 15th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM)*, Davis, USA, November 2004.

[11] E. Adar and B. A. Huberman, " Free Riding on Gnutella," Tech. Rep., HP, 2000, http://www.hpl.hp.com/research/idl/papers/gnutella/.

[12] K. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and analysis of a Peer-to-Peer File Sharing Workload," in *The 19th ACM Symposium on Operating Systems Principles*, New York, USA, October 2003, pp. 314–329.

[13] "eBay Feedback," http://pages.ebay.com/help/feedback/reputation-ov.html.

[14] S. Marti and H. Garcia-Molina, "Limited Reputation Sharing in P2P Systems," in *ACM Conference on Electronic Commerce*, New York, USA, May 2004, pp. 91–101.