Journal of Next Generation Information Technology
Volume 1, Number 1, May 2010

Towards Trustworthy Peer-To-Peer Environments: An Appraisal Analysis Approach

Farag Azzedin and Salman Khwaja
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
Email: {fazzedin,khwaja}@kfupm.edu.sa
doi: 10.4156/jnit.vol1.issue1.6

Abstract

The recent and rapid development of P2P systems is paving the way for a distinguished manner of computing where autonomy and decentralization are the foundation of success. In this paper, we focus on the problem of free riding in decentralized collaborative environments. We propose a novel free riders filtering algorithm based on trustworthiness. Existing, literature does not consider trustworthiness, which we believe is a vital dimension that should be considered when identifying free riders. Our proposed algorithm is based on appraisal analysis to filter out and isolate free riders. Our extensive simulation experiments show that our proposed algorithm is reasonably successful in identifying free riders in P2P systems.

Keywords: *P2P systems, Trust, Reputation*

1. Introduction

Peer-to-Peer (P2P) computing systems have emerged as an important paradigm for distributed computing due to their potential for the involvement of nodes at a large-scale for the purpose of sharing and collaboration. P2P systems facilitate direct resource sharing among dynamic peers and are characterized as being decentralized, self-organizing distributed systems that exploit and efficiently make use of the untapped resources of heterogeneous hosts. A pure P2P file transfer network does not have the notion of clients or servers, but only equal peer nodes that simultaneously function as both ‘‘clients’’ and ‘‘servers’’ to other nodes on the network. This model of network arrangement differs from the client–server model where communication usually utilizes a central server [6].

With the recent and rapid development of P2P systems, such P2P systems attract the attention of researchers, but they also bring up some problems and concerns. Some peers might be buggy and cannot provide services as they advertise. Some might be malicious by providing bad services to get more benefit. Since there are no centralized nodes to serve as an authority to supervise peers’ behaviors and punish peers that behave badly, malicious peers can get away with their bad behaviors. How to distinguish potential benevolent peers from potential buggy or malicious peers is the current research hot spot of P2P system. If these problems were resolved, the P2P system would have more applications.

At present the main method of resolving these problems is to build trust and reputation in the system. The trust and reputation mechanism is derived from human society. Trust is at the core of most relationships between human beings. Take the simple example of purchasing an item from a shop. We may choose to buy a certain brand because we have found it to be trustworthy in the past or it has a reputation for being widely “trusted”.

Trust has important role in the Semantic Web, as agents need to make trust judgments when alternative sources of information are available. Computers will have the challenge to make judgments in light of the varying quality and truth that these diverse “open” (unedited, uncensored) sources offer. Today, web users make judgments routinely about which sources to rely on since there are often numerous sources relevant to a given query, ranging from institutional to personal, from government to private citizen, from objective report to editorial opinion, etc [5].

Reputation systems [3, 7] provide a way for building trust through social control without trusted third parties. Most research on reputation-based trust utilizes information such as community-based feedbacks about past experiences of peers to help making recommendation and judgment on quality and reliability of the transactions. Community based feedbacks are often simple aggregations of positive Proceedings of the negative feedbacks that peers have received for the transactions they have performed and cannot accurately capture the trustworthiness of peers. In addition, peers can misbehave in a number of ways, such as providing false feedbacks on other peers. The challenge of building a trust mechanism is how to effectively cope with such malicious behavior of peers [4].

Another challenge is that trust context varies from transactions to transactions and from communities to communities. It is important to build a reputation based system that is able to adapt to different configurations and different situations. Furthermore, there is also a need for experimental evaluation methods of a given trust model in terms of the effectiveness and benefits. Most traditional trust models only give an analytical model without any experimental validation due to the subjective nature of trust. There is a need of general metrics for evaluating the effectiveness and benefits of trust mechanisms.

The development of any complex computer architecture can be a challenge. This is especially true of a complex distributed algorithm that is run by autonomous untrusted agents, yet is expected to be relatively reliable, efficient, and secure. Such is the task of designing a complete reputation system for use in peer-to-peer networks [2].

2. Literature Review

Many have tried to compute the value of trust by modeling and reasoning. A wide variety of literature now exists on trust, ranging from specific applications to general models. The meaning of trust as used by each researcher differs across the span of existing work. We here provide three general definitions from existing research. The first definition, from Mui et al. [11], refers to past encounters, and it can be taken as "reputation-based" trust "[Trust is] a subjective expectation an agent has about another's future behavior based on the history of their encounters."

Another definition, from Grandison and Sloman [12], introduces context and is unique. It refers to the "competence" to act (instead of actions, themselves) "[Trust is] the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context."

The next definition, from Olmedilla et al. [13], is again very general, and it refers to actions and not competence like the previous definition "Trust of a party A to a party B for a service X is the measurable belief of A in that B behaves dependably for a specified period within a specified context (in relation to service X)."

A crux of these definition is that trust is only worth modeling when there is a possibility of deception, that is, when there is a chance of a different outcome than what is expected or has been agreed upon. In the view of Aberer, trust is a crucial prerequisite or performing business and cooperation in P2P environments due to the autonomous peers and dynamic connections [4]. Trust management is defined as a mechanism that enables peers to establish mutual trust relationship in order to improve the success rate of cooperation and transactions among peers.

Donovan et al. have organized trust research in four major areas [14]. First, policy-based trust that uses policies to establish trust, focused on managing and exchanging credentials and enforcing access policies. Work in policy-based trust generally assumes that trust is established simply by obtaining a sufficient amount of credentials pertaining to a specific party, and applying the policies to grant that party certain access rights. Second, reputation-based trust that uses reputation to establish trust between different entities. Here, past interactions or performance for an entity are combined to assess its future behavior. Research in reputation-based trust uses the history of an entity's actions/behavior to compute

trust, and may use referral based trust (information from others) in the absence of (or in addition to) first-hand knowledge. Third, general models of trust. There is a wealth of research on modeling and defining trust, its prerequisites, conditions, components, and consequences. Trust models are useful for analyzing human and non-human trust decisions and for utilizing computable models of trust. Fourth, trust in information resources. Trust is an increasingly common theme in Web related research regarding whether Web resources and Web sites are reliable. Moreover, trust on the Web has its own range of varying uses and meanings, including capturing ratings from users about the quality of information and services they have used, how web site design influences trust on content and content providers, propagating trust over links, etc.

As we are dealing with P2P based trust model, so we find another categorization which is done by Qiao et al [15]. They have divided P2P reputation model into three categories: PKI-based reputation model [16], local reputation model [17], and global reputation model [18]. A number of reputation systems and mechanisms are proposed for online environments and agent systems in general [19, 20]. Most of them assume the feedback is always given honestly and with no bias and paid little attention to handle the situation where peers may conspire to provide false ratings or dishonest recommendation.

There are proposals which have attempted to address the issue of quality of the feedbacks. One such proposal for computing and using reputation for Internet ratings by Chen et al. [21] differentiates the ratings by computing a reputation for each rater based on the quality and quantity of the ratings it gives. This method is using the assumption that the ratings are of good quality if they are consistent to the majority opinions of the rating. So this method can easily be cracked by adversaries. Adversaries who submit fake or misleading feedbacks can still gain a good reputation as a rater in this method simply by submitting a large number of feedbacks and becoming the majority opinion.

Another mechanism proposed by Dellarocas [22], combats two types of cheating behavior while submitting feedbacks. The idea behind is to detect and filter out exceptions in certain scenarios using cluster-filtering techniques. This technique can be applied into feedback-based reputation systems to filter out the suspicious ratings before the aggregation.

In comparison, Xiong et al. trust model is more general [23]. They use the credibility of the feedback source as one of the basic trust parameters when evaluating the trustworthiness of peers. The credibility factor can be also used to detect fake or misleading ratings.

Singh & Liu shows the importance of anonymity in decentralized P2P systems. They propose a TrustMe protocol to support mutual anonymity in both data providers and data requesters [24]. TrustMe use public key cryptography schemes to guarantee security and resist attacks. It features anonymity, reliability and accountability.

Xiong and Liu present a reputation-based trust framework PeerTrust to qualify the trustworthiness of peers [25]. They think that a trust framework that is merely based on previous transaction feedback is insufficient and inaccurate. PeerTrust includes five important factors to evaluate the trustworthiness for each peer: (1) the feedback obtained from other peers; (2) the feedback scope; (3) the credibility factor for the feedback source; (4) the transaction context factor; (5) the community context factor. In this way, PeerTrust can evaluate peers from multiple facets.

Zhuge & Chen proposed a trust-based probabilistic search mechanism P-Walk to guide neighbor selection when routing in Gnutella-like unstructured P2P networks [26]. P-Walk combines trust feedback mechanism and probability mechanism. Its unique characteristic is that peers will preferentially deliver query messages to those who they think most likely to return desirable data. Simulation results show that P-Walk can not only improve query recall and precision, but also can reduce the traffic cost and alleviate the malicious behaviors.

Hou et al. claim that current trust models can be roughly classified into two types: partially peer reputation model that only collects local trust feedbacks, and the wholly peer trust model that collect

global feedbacks. They provide a global trust model based on confirmation theory. Peers' trust values are computed by C-F model based on the bartering experience, and stored in a structured P-Grid [27].

Ries proposes a new trust model CertainTrust to help peers to find reliable partners in a decentralized manner [30]. J.Golbeck et al. build a trust model on the semantic web by applying the theory of the social network into the semantic network [29]. G. Suryanarayana et al. present two file-sharing application prototypes for P2P networks by integrating XREP with the PACE architecture [31]. Poblano is a distributed Trust model build on top of JXTA platform [28]. It can be applied in the reputation-based search and recommendation systems.

3. Problem Statement

The development of any complex computer architecture can be a challenge. This is especially true of a complex distributed algorithm that is run by autonomous untrusted agents, yet is expected to be relatively reliable, efficient, simple and secure. This is the task of designing a trust filtering system for use in peer-to-peer networks which we are trying to accomplish in this research. Each algorithm is designed for some specific configurations and settings. Similarly our Trust filtering algorithm is also designed with specific set of configurations in mind. So we have few assumptions for our algorithm which are required for getting optimal performance from our algorithm. First assumption is that we are assuming that peer don't forge their ID's. Once a peer has got an id it will stay with it. Also, peers can be malicious or untrustworthy but the network is secure. So if a peer gives any feedback, it reaches as is.

4. Proposed Solution

Most of the existing algorithm use recommendations to find trustworthy peers. Although it is a good technique but it fails when majority of the peers are dishonest. As its been discussed that honesty and trustworthiness are separate properties therefore even when there are many trustworthy peers in the system, the dishonest peers will stop the user from transacting with them.

In our proposed algorithm, we bypass this added layer of honesty checking and tried to check the trustworthiness of the peers. In our proposed algorithm we have two lists, a) black list, which is private to each peer and in which all the peers are marked with which the peer has an experience of untrustworthy transaction, b) Appraisal List, which is global and in which a peer appraise the target peer after it has a successful transaction. The peer can only appraise a target peer once in one time slot. A single time slot is a specific duration of time in which all the appraisals will be marked with the same time id.

In our proposed algorithm the source peer can only appraise the target peer and no complaint can be filed against the target peer. The source peer can put the untrustworthy peers in its black list so that it can avoid them in the future. But this black list is not shared by other peers. Each peer has its own black list. Our proposed algorithm's pseudo-code is shown below.

Our Algorithm starts by declaring all the variables, some of which are mentioned in lines 1–4. The first loop in our code which is shown in lines 5–15 is used for adding some data in Appraisal list. This Appraisal List is global and will be accessible to everyone in the system. So that each source Peer can use the data from the Appraisal List for their own decision making.

The second for loop from lines 16-29 is used for simulation. After selecting the source and target peer randomly we first check if the target peer is in source peer's black list or not. If it is in source peer black list we simply cancel the transaction else we see how many Appraisal the target peer got in last time slot and also how many appraisal the target peer got in before last time slot which is current time slot minus 2. After getting these two numbers we see if the number of appraisals in last time slot is greater than two third of the appraisals in the time slot before it. This condition is shown in line 24. If the condition is true than the source peer will perform the transaction else it will cancel the transaction.

AC is the activity constant which is taken from 0 to 1. We have used the value 0.66, because mostly 0.66 is considered as a constant which is used most in the literature and also we found it in our simulation to provide optimal results.

```
1 Set DishonestPeersRatio
2 Set UntrustworthyPeerRatio
3 Create AppraisalList
4 Create BlackList
5 For i:1 to m
6       // warm up the system with entries in AppraisalList
7       Select SourcePeer randomly
8       Select targetPeer randomly
9       //Perform transactions
10 If (transaction is a trustworthy transaction) then
11      Add Appraisal
12 Else
13      Add to Personal Black List
14 End If
15 End For
16 For  i:m to n
17      Select SourcePeer randomly
18      Select TargetPeer randomly
19      If (In Personal blacklist)
20            Cancel transaction
21      Else
22            NA(t - 1) = number of Appraisals in last time slot
23            NA(t - 2) = number of Appraisals in last time slot
28      End If
24      If (NA(t - 1) > NA(t - 2) x AC) Then
25            Do transaction
26      Else
27            Cancel Transaction
28      End If
29 End For
```

Figure 1: Pseudo code for Appraisal Analysis Algorithm.

5. Performance Evaluation

5.1. Simulation Setup

In simulation, we selected Credibility Factor (CF) [38], Modified Outlier Filtering (MOF) [39] and also results without using any filtering (WF). WF is used as the baseline for the results.

For simulation we used 1024 peers. Each simulation was performed 10 times and average was taken of it. With the size of 1024 peers we get the confidence interval of 3.05% and 4.01% at confidence level of 95% and 99% respectively, when the population size is 118,925 nodes, which was the size of Gnutella networks in February 2006. If we increase the population size to 1 million nodes, the confidence interval changes slightly to 3.07% and 4.03% with the confidence level of 95% and 99% respectively.

Each peer performed an average of 80 transactions per simulation run. Initial 10 transactions are used for warming up the system and filling the Appraisal list. Each transaction's source peer and target peer were randomly generated. A source peer will commit a transaction if it predicts that the target is trustworthy. After committing a transaction the source peer can provide appraisal. If the source peer is honest and the transaction was trustworthy it will give an appraisal and if the transaction was untrustworthy it will not give an appraisal. If the source peer is dishonest and the transaction committed was trustworthy it will not provide an appraisal but when the transaction is untrustworthy it will give an appraisal.

For simulation and checking the performance of our algorithm we used different proportions of trustworthy and untrustworthy peers. The ratio of the trustworthy peer was set at 30% and then 60%. Similarly the ratio for the dishonest peer was set from 20%, 40%, 60% and 80%. The Activity constant value was set at 0.66%.

5.2. Verification of Activity Constant value

As mentioned earlier in the test setup, that we fixed the Activity Constant value to 0.66. In this section we are going to see the effects of changing the Activity Constant value from 0.66 to 0.8 and also at 0.50. First we will see the effect of changing the Activity constant value on the percentage of the committed trust worthy transaction.

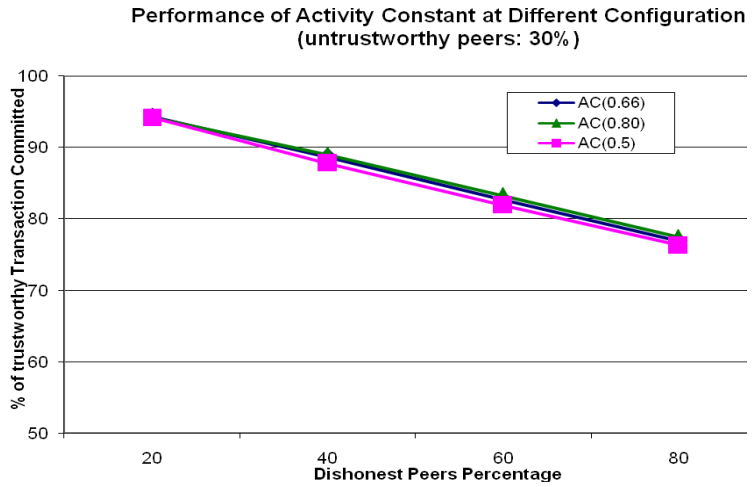

Figure 2: Effect of Changing the Activity Constant Value on the Performance

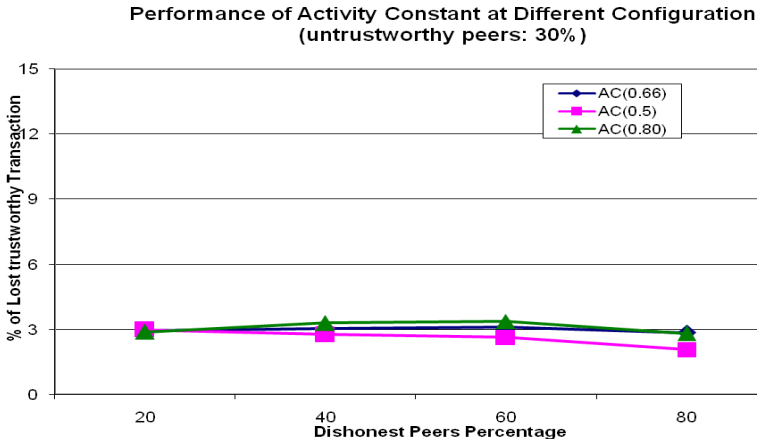

Figure 3: Effect of Changing the Activity Constant Value on the Performance

We see in the Figure 2 and Figure 3 that changing the Activity constant value does not have much profound effect on the results. Interesting observation in the graph is that the effect of Activity Constant is more when the dishonest peers are high (80%). The reason is that when the dishonest peers are high, the numbers of appraisals are going to be low and changing the Activity Constant will impact the decision more.

The result shows that the activity constant value of 0.5 perform better in percentage of lost trust worthy transaction as it allows more transaction to be committed, so total number of trustworthy transactions lost will be very few. But it performs worst in percentage of committed trustworthy transaction as low threshold for the transaction to be committed will also add some malicious transactions. So activity factor of 0.5 increases recall but drop precision.

The activity constant value of 0.8 perform better in the percentage of committed trustworthy transaction but it perform slightly behind in percentage of percentage of trustworthy transactions lost. The result shows that there is not a lot of difference between the activity constant value of 0.5, 0.66 and 0.8.

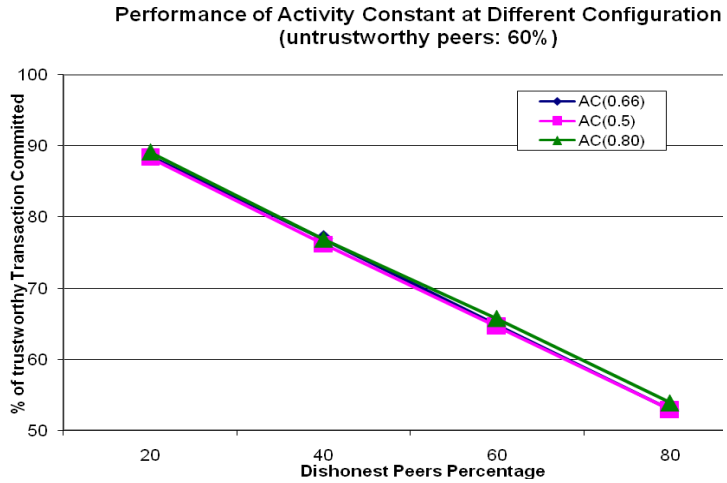

Figure 4: Effect of Changing the Activity Constant Value on the Performance

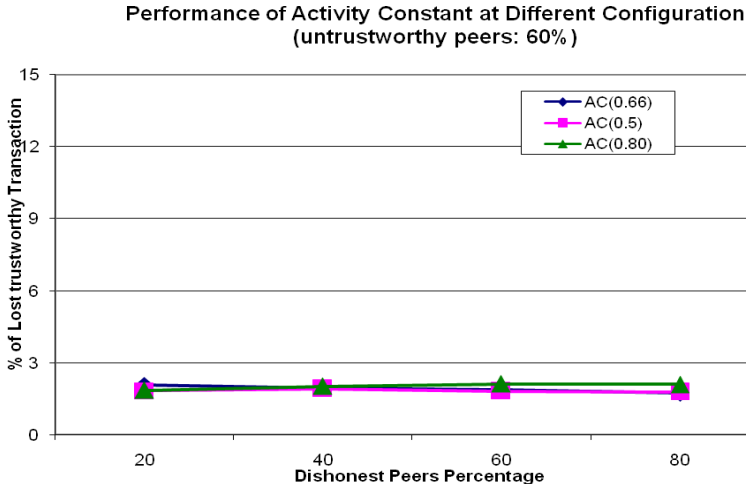

Figure 5: Effect of Changing the Activity Constant Value on the Performance

Again we see a similar pattern in Figure 4 and Figure 5 in which the percentage of untrustworthy peers is 60%. The 0.5 activity constant performance slightly better in % of lost trustworthy transaction and activity constant of 0.8 just performing best in percentage of trust worthy transactions committed.

Although there is not a great deal of difference between three Activity Constant values but we choose the value of 0.66. The reason for this is that this value performs in the optimal in all the tests. So we can say that Activity Constant of value 0.66 provides optimal output for all the configuration and setups. If we decrease the Activity Constant value the recall increases but it also slightly decrease the precision and the percentage of trustworthy transaction committed decreases. Similarly if we increase the Activity constant value it will decrease recall so the overall system will loose more trustworthy transactions but the peer itself will make more trustworthy transactions. So the Activity constant value can be modified according to the requirement of the system or peer. Now we start our main comparison by checking the performance of the algorithm in performing trustworthy transaction. The numbers of untrustworthy peers are at 30%.

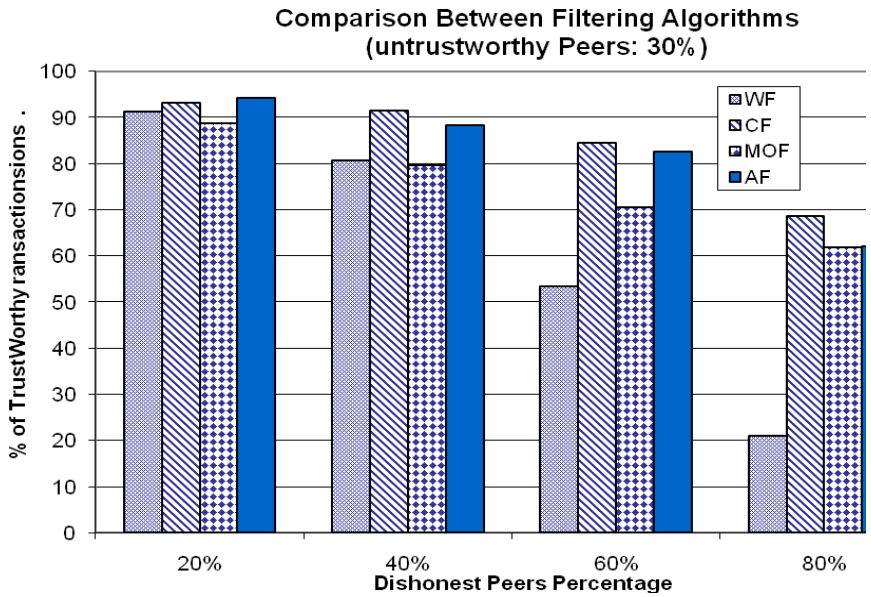

Figure 6: Performance of Different Trustworthy Peer detection Algorithms

We see that our proposed Activity Factor algorithm performs at above 90% when the dishonest peers are 20% (Figure 6). The performance remains above 80% even when the dishonest peers are increased to 60%. The reason for this high performance is that we are directly evaluating the trust worthiness of the target peer by checking the appraisals the target peer achieve in a particular time slot. This approach is much simpler than other approaches which first gather recommendations from other peers and then perform recommender's evaluation and then they try to figure out the trustworthiness of the target peer through the valid and honest recommenders. This extra step of evaluating the recommenders is equally complex and with little return in increasing the performance of the algorithm.

Moreover as we mentioned in Our Approach Section that we have black list also and the source peer whether honest or dishonest is not going to commit a transaction with a target peer which is in the black list. Therefore even the dishonest peer after transacting and giving appraisal to untrustworthy peer is not going to transact again with the untrustworthy peer in its black list. The performance still is not able to match the credibility factor as it is keeping a separate vector for every peer's honesty. So the peer has a very detailed history of every other peers honesty and trustworthiness. This makes it much more complicated.

Another point to note in the result is that Activity Factor is better than Modified outlier filtering with 40% and 60% dishonest peers. There are two reasons for it, firstly modified outlier filtering is not

very good when the numbers of honest and dishonest peers are around same, because then there are no clear cut outliers. There will be two groups of almost equal recommender peers and one will be honest and other will be dishonest. So it will be hard for the modified outlier filtering to detect from it. Secondly although Activity factor try to restrict colluding by imposing a limit of 1 appraisal per time slot, still with 80% dishonest peers the amount of peers colluding will be very high.

Secondly we check the performance of the algorithm in the number of trustworthy transactions lost (Figure 6). This is an important metric and allows us to see that if the algorithm discards too many trustworthy transactions in trying to keep the percentage of trustworthy transaction committed high. Again the percentage of the untrustworthy peers is set to 30%.

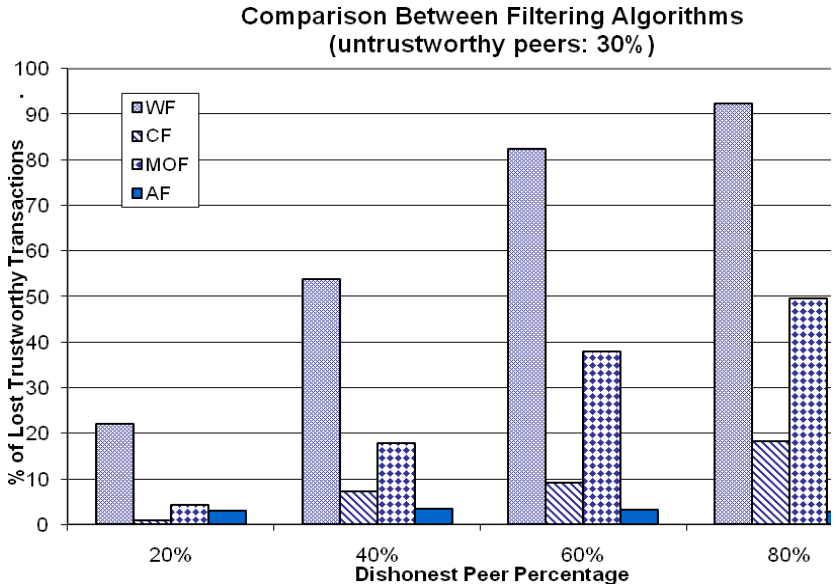

Figure 7: Performance of different trustworthy peer detection algorithm

In this case we see that the activity factor performs best. As we stated earlier that in Activity Factor there is no bad mouthing therefore in the lost trustworthy transaction we see that there is virtually no trustworthy transaction is lost. . The reason for it is simple, in other algorithm the dishonest peers can give false complains or badmouth to stop the trustworthy peers getting transactions. So if a trustworthy peer has few appraisals from honest peers and many bogus dishonest complaints than the algorithm will not allow the peer to transact but in Activity factor if the number of appraisals remains above the activity constant threshold, then trustworthy peer will keep getting transactions. This is a very good sign for the service provider, as they can see from these results that they will not loose transaction under our algorithm.

Next up we increase the number of untrustworthy peers and check the performance of our algorithm. We want to see if our algorithm will work even in an environment where untrustworthy peers are in dominant or in other word how our algorithm will perform in an untrustworthy environment. The percentage of untrustworthy peers is set at 60.

We see in Figure 8 that the performance of the algorithm has reduced but not by much. The algorithm perform almost 90% trustworthy transactions with 20% dishonest peers, the performance drops to just below 80% with 40% dishonest peers and it finally reached around 55% with 80% dishonest peers. The important thing to see here is that unlike other algorithm the performance drop is linear and never goes below 50%. The credibility factor still performs best due to the same reason that we have very detailed individual data with each peer about every other peer's honesty and

trustworthiness. The performance of the modified outlier filtering is a lot more erratic at 40, 60 and 80% of dishonest peers.

As the percentage of untrustworthy peer has been increased to 60% therefore the chances for the dishonest peers to collude has also increased, therefore showing a bit of decrease in the performance of Activity factor in comparison to 30% untrustworthy peer result. Next we check the percentage of lost trustworthy transaction when the percentage of untrustworthy peers is set to 60%.

Again in Figure 9 we see that the performance of the Activity factor is very impressive. The Activity factor performs best among all the algorithms. The percentage of the lost trustworthy transactions stays between 5%. This means that through the use of our algorithm the trustworthy peers will almost always get a transactions no matter how high the percentage of dishonest peers get.

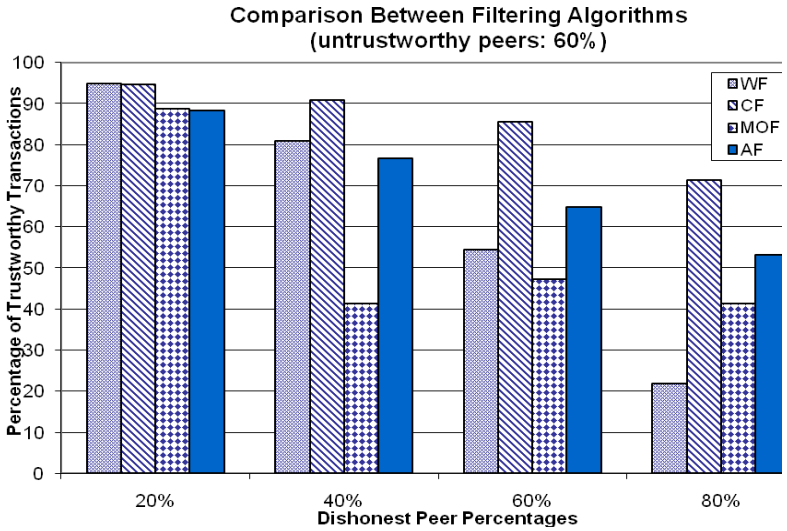

Figure 8: Performance of different trustworthy peer detection algorithm

5.3. Impact of Collusion

In activity factor we tried to restrict colluding by restricting each peer to give almost 1 appraisal per time slot to the target peer. So anytime a source peer tries to give second appraisal to a target peer within the same time slot than it will not be allowed by the system.

To check whether this really provide any benefit, we devised a small simulation for it. In this simulation we had 256 peers and each peer performed on average 200 transactions. Along with the peers we also created colluding groups. Each group consists of almost 8 peers. The only dishonest peers were allowed to be part of the colluding group. Peer within the colluding group always appraised each other. Also the colluding peers were given twice as many instructions as the normal peer, because colluding peers are there for increasing the appraisals of the colluding peer. In the first setup we applied the restriction of one appraisal per time slot for each target peer by the source peer.

In the second setup same configuration were used but the restriction of one appraisal per time slot for each transacted target peer by the source peer was not implemented. We wanted to see how many times a peer is detected by the algorithm for giving a second appraisal to the target peer and how much increase in untrustworthy transaction does it cause. So in this simulation we are only concerned with the decrease in trust worthy transaction committed and the percentage of colluding detected. Here we will also like to mention that these results are not comparable with earlier results as the configuration has been changed and also the time has been adjusted a little. First we see the result of the simulation in trustworthy environment, with untrustworthy peers at 30%.

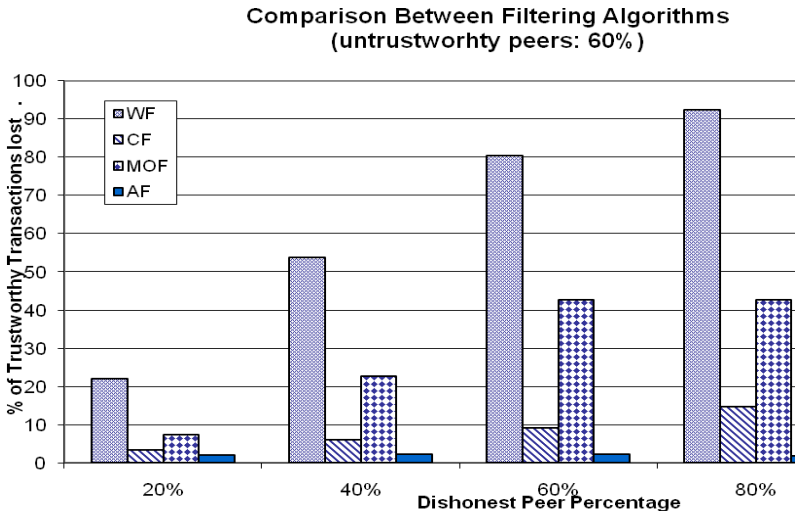

Figure 9: Performance of different trustworthy peer detection algorithm

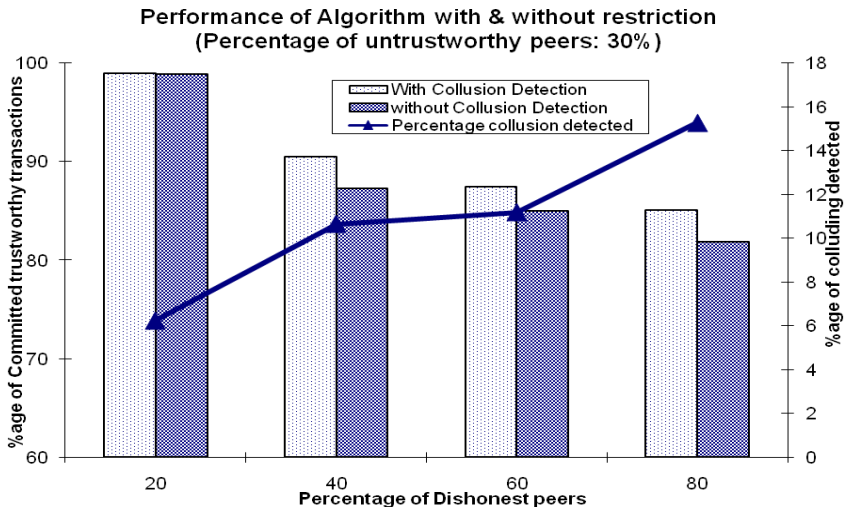

Figure 10: Collusion detection & its effect on Performance

When the percentage of dishonest peers is less than 50%, we have used 2 colluding groups. We see that at 20% dishonest peer we detected only around 6% colluding and its effect on percentage of committed trustworthy transaction is even less than 1%. But when we increase the dishonest peers the percentage of colluding detected is around 10% – 15% and it bring 3% – 5% improvement in percentage of trustworthy transactions committed. Next we check the result in the untrustworthy environment, where the percentage of untrustworthy peers is 60%.

In untrustworthy P2P environment we again see similar trend where with the increase of dishonest peer the percentage of detected collusion has increased (figure 11). For 20% dishonest peers the improvement in committed trustworthy transaction is around 1%. But as we increase the dishonest peers percentage the improvement in percentage of trustworthy transactions committed also increases to around 5%.

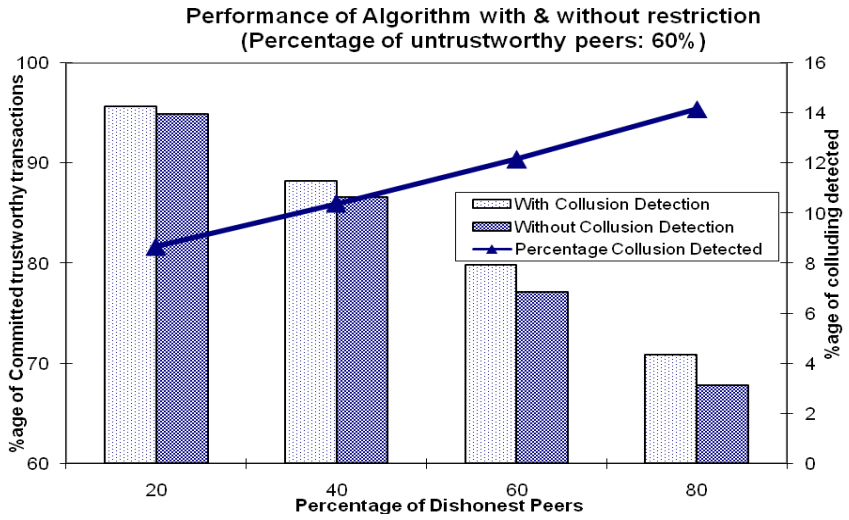

Figure 11: Collusion detection & its effect on Performance

5.4. Attack Resilience

In this section we list the most common attack types and exploits which are used by the malicious peers to harm other peers. And show how our proposed algorithm countermeasure such attacks.

5.4.1. Bad mouthing/Slandering

As long as recommendations are taken into consideration, malicious parties can provide dishonest recommendations [8] to frame up good parties. This attack is referred to as the bad mouthing attack. This attack has been discussed in many existing trust management or reputation systems [9], [8]. As mentioned earlier, our algorithm does not give the peer to complain. So there is no chance for any peer to badmouth against any other peer.

5.4.2. Collusion

In many situations multiple malicious peers acting together can cause more damage than each acting independently. This is especially true in peer-to-peer reputation systems, where covert affiliations are untraceable and the opinions of unknown peers impacts ones decisions. Most research devoted to defeating collusion assume that if a group of peers collude they act as a single unit, each peer being fully aware of the information and intent of every other colluding peer [10].

Our algorithm does have the possibility of being exploited by the colluding. But in our algorithm we have tried to minimize the chances of colluding by firstly allowing the source peers to appraise only once in a single time slot. Also back to back appraisals are not allowed in our approach. So this enhances the algorithm resilience against the small group of colluding peers.

5.4.3. Sybil Attack

Peer-to-Peer (P2P) networks are logical or virtual networks on top of already existing networks, i.e., they are mostly built upon an underlying network like the Internet. Most P2P networks use their own ‘addressing scheme’ based on logical identifiers for structuring and organizing the network. A standard requirement for the identifier assignment procedure says that node identifiers should be globally unique [33].

When an entity can run a large number of nodes and obtain a large number of node identifiers, the P2P network can be dominated by this entity. This dominance can be used to undermine replication mechanisms, which results in less robustness. Finally one entity can control the whole P2P network. This is commonly known as Sybil attack [32]. This attack is also possible. But the chances of it are minimized by using analyzing the variation in the appraising peer set for the target.

5.4.4. Self Promoting

Self promoting is also related to giving false feedback [35]. In self-promoting, the attackers only increase their own feed-back reputation. In this way they try to minimize the effect of true negative feedback given to them. This attack is not possible as the appraising list keeps the target peer and source peer id. Moreover source and target peer ids need to be different for the appraisal to be accepted.

5.4.5. RepTrap

Attackers find some high quality objects that have a small number of feedbacks. Then, attackers provide a large number of negative feedbacks to these objects such that these objects are marked as low-quality by the system. That is, the system makes wrong judgment about the object quality. As a consequence, the system thinks the attackers' negative feedbacks agree with the object quality and the feedbacks from honest users disagree with the object quality. Therefore, the feedback reputation of the attackers will be increased while the feedback reputation of honest users will be reduced.

RepTrap is applicable to reputation systems that calculate key reputation scores based on feedbacks as well as metrics describing whether users provide honest feedbacks. These metrics are often referred to as feedback reputation. Compared with other known attacks that achieve the similar goals, the RepTrap requires less effort from the attackers and causes multi-dimensional damage to the reputation systems [34]. This attack is also not possible as a peer cannot lower the other peers feedback by giving false untrustworthy rating. As no complaints can be filed so the trustworthy peers are safe from RepTrap.

5.4.6. White-washers

Peers that purposefully leave and rejoin the system with a new identity in an attempt to shed any bad reputation they have accumulated under their previous identity [36]. When a peer leaves and rejoins a system with new identity than there is no local or global reputation history available for that peer. When no information can be located, an agent must decide whether to transact with a stranger based on its stranger policy. As mentioned previously, two simple strategies are to optimistically trust all strangers, or pessimistically refuse to interact with them. Both have their drawbacks. Optimistic agents may frequently be defrauded, especially in systems with high levels of whitewashing. However, in pessimistic systems, new users will be unable to participate in transactions and will never build a reputation [2].

This attack only happens when the peer rejoin the system with a new identification. As we are assuming that the peer can get only one unique identification (see Section 4), therefore this attack is beyond the scope of this research.

5.4.7. Impersonation

Impersonation refers to the threat caused by a peer that portrays itself as another in order to misuse the target peer’s reputation or privileges [37]. Impersonation also falls under identity management which is not covered in this research.

5.4.8. Ballot stuffing

When a single malicious user can issue an unlimited number of "good" votes for raising the feedback rating of colluding users or can issue "bad" votes for demoting the feedback rating of competing users [38]. This is known as ballot stuffing. Each appraisal given is tied with a valid transaction, so the ballot stuffing is also not possible in this paper.

5.4.9. Free-Riders

Non-collaborative users, who do not contribute to the community, are referred to as free riders in the literature. The phenomenon of free riding does not just create a tedious task that can be bothersome for the contributors, but also can create a potential performance bottleneck. As a consequence, responses to queries can experience a longer delay. In a collaborative file sharing system, 66% of users do not share files (i.e., free riders) and the top 25% users account for 99% of all downloads. Free riding must be tackled before P2P-based technology can be used to provide the necessary substrate to multimedia applications and services so that they can be redeployed over P2P networks. Free riders can also be easily identified with our proposed algorithm, as each appraisal has a source and target peer, so the peers who do not have any appraisal in the list can easily be identified as free riders.

6. Conclusions and Future work

In this paper, we devise a novel algorithm for the detection of trustiness of the target peer which is simple to implement and provide great resilience against number of attack types. While analyzing the existing trust detection algorithms, we found that they all rely on the honest opinion of the recommenders. As the number of dishonest increases, the detection rate of these algorithms also starts decreasing rapidly. Secondly looking at the attack types in the exiting trust detection algorithms, we saw that the majority of these types of attacks are concerned with the false untrustworthy rating. So while devising the new algorithm for trust detection, we aimed at minimizing our dependency on the recommender peers and also tried to remove the option of negative feedback altogether that adds an overhead layer to the trust system.

Although colluding can still be possible but it has been minimized as shown in the simulation results. Furthermore, collusion can be further minimized by checking the uniqueness of the appraisal giver for a target peer. If the appraisal giver is unique or different over a certain threshold, then we can say that target peer is not colluding. Similarly, we can have an internal list of colluding peers. So, whenever we make an untrustworthy transaction we put all those peers which appraised the target peer in that internal colluding peers list. As such, if we find these peers in another target peer appraisal list we don't commit the transaction.

Currently, we are checking two time slots to make a decision about a target peer. We can increase the number of time slots we are checking for the target peer and thus can make a more informed decision based on the trend spread over more number of time slots. Similarly we can see the percentage of appraisal the target peer is getting in the system in each time slot. This will help us see if the number of appraisal the peer is getting are increasing with the growth of the system.

7. References

[1] M.S. Hou, X.L. Lu, X. Zhu, and C. Zhang, "A trust model of P2P system based on confirmation theory", ACM SIGOPS Operating Systems Review, ACM Press, 2005, pp. 56-62.
[2] Sergio Marti, Hector Garcia-Molina, Taxonomy of trust: Categorizing P2P reputation systems in Computer Networks 50 (2006), 2006, 472–484
[3] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. Communications of the ACM, 43(12), 2000.
[4] Li Xiong, Ling Liu, A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities, in Proceedings of the IEEE International Conference on E-Commerce (CEC'03), 2003

[5] Donovan Artza, Yolanda Gil, A survey of trust in computer science and the Semantic Web in Elsevier B.V., 2007
[6] Raouf Boutaba and Alan Marshall. Management in peer-to-peer systems: Trust, reputation and security. Computer Networks, 50(4):469–471, 2006.
[7] C. Dellarocas, "Mechanisms for coping with unfair ratings and discriminatory behavior in online reputation reporting systems," in Proceedings of ICIS, 2000.
[8] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in Proceedings of 12th International World Wide Web Conferences, May 2003.
[9] P. Maniatis, M. Roussopoulos, T. Giuli, D.S.H. Rosenthal, M. Baker, Y. Muliadi, Preserving peer replicas by rate-limited sampled voting, in: 19th ACM Symposium on Operating Systems Principles (SOSP 2003), 2003.
[10] L. Mui, M. Mohtashemi, A. Halberstadt, A computational model of trust and reputation, in: Proceedings of the 35th International Conference on System Science, 2002, pp. 280–287
[11] T. Grandison, M. Sloman, A survey of trust in internet applications, IEEE Commun. Surv. Tutorials 4 (4) (2000) 2–16.
[12] D. Olmedilla, O. Rana, B. Matthews,W. Nejdl, Security and trust issues in semantic grids, in: Proceedings of the Dagsthul Seminar, Semantic Grid: The Convergence of Technologies, vol. 05271, 2005.
[13] K. Aberer, Z. Despotovic, Managing Trust in a Peer-to-Peer Information System, in Proc. of 10th Int'l. Conf. on information and knowledge management (CIKM), 2001, 310-317.
[14] Donovan Artz, Yolanda Gil, A survey of trust in computer science and the Semantic Web, in: Web Semantics: Science, Services and Agents on the World Wide Web 5, (2007), 58–71.
[15] Shaojie Qiao, Xingshu Chen, Changjie Tang, ThTrust: Transaction History Based Peer-to-Peer Trust Model, First International Symposium on Data, Privacy and E-Commerce.
[16] K. Berket, A. Essiari, and A. Muratas, "PKI-based security for peer to peer information sharing", Proceedings of the Fourth IEEE International Conference on Peer to Peer Computing, 2004, pp. 45-52.
[17] F. Cornelli, E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati, "Choosing reputable servents in a P2P network", Proceedings of the Eleventh International World Wide Web Conference, Honolulu, Hawaii, 2002, pp. 441-449.
[18] S.D. Kamvar, and M.T. Schlosser, and H. G. Molina, "The EigenTrust algorithm for reputation management in P2P networks", Proceedings of the 12th International Conference on World Wide Web, Budapest, Hungary, 2003, pp. 640-651.
[19] B. Yu and M. P. Singh. A social mechanism of reputation management in electronic communities. In Cooperative Information Agents, 7th International Conference, CoopIS, 2000.
[20] G. Zacharia and P. Maes. Trust management through reputation mechanisms. Applied Artificial Intelligence, 14(8), 2000.
[21] M. Chen and J. P. Singh. Computing and using reputations for internet ratings. In 3rd ACM Conference on Electronic Commerce, 2001.
[22] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In 2nd ACMConference on Electronic Commerce, 2000.
[23] A. Singh, L. Liu, TrustMe: anonymous management of trust relationships in decentralized P2P system, in Proc. 3rd Int. Conf. on Peer-to-Peer computing, 2003, 142-149.
[24] L. Xiong, L. Liu, PeerTrust: supporting reputation-based trust for Peer-to-Peer electronic communities, IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE), 16(7), 2004, 843-857.
[25] H. Zhuge, X. Chen, X. P. Sun, Trust-based probabilistic search with the view model of peer-to-peer networks. Concurrency and Computation: Practice and Experience, 18(14), 2006, 1839-1855.
[26] M. S. Hou, X.L. Lu, X. Zhou, C. Zhan, A Trust Model of P2P System based on Confirmation Theory, ACM SIGOPS Operating Systems Reviews, 39(1), 2005, 56-62.
[27] R. Chen and W. Yeager, Poblano: A Distributed Trust Model for Peer-to-Peer Networks, Technical Report, TR-14-02-08, Palo Alto, Sun Microsystems, 2002, http://gnunet.org/papers/jxtatrust.pdf.

[28] J. Golbeck, B. Parsia, and J. Hendler, Trust Networks on the Semantic Web, in Proc. Cooperative Information Agents, 2003, 238-249.
[29] S. Ries, Certain Trust: a Trust Model for Users and Agents, in Proc. ACM Symposium on Applied Computing, Seoul, Korea, 2007, 1599-1604.
[30] G. Suryanarayana, M. H. Diallo, J. R. Erenkrantz, R. N. Taylor, Architecting trust-enabled Peer-to-Peer File-Sharing Applications, Crossroad, 12(4), 2006, 5.
[31] J. Douceur. The Sybil Attack. In 1st International Workshop on Peer-to-Peer Systems (IPTPS '02). Springer,, 2002.
[32] J. Dinger and H. Hartenstein, "Defending the Sybil Attack in P2P Networks: Taxonomy, Challenges, and a Proposal for Self-Registration", In ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security (ARES'06), pp. 756–763, 2006.
[33] Yafei Yangy, Qinyuan Feng, Yan Lindsay Suny, Yafei Dai, RepTrap: A Novel Attack on Feedback-based ReputationSystems, in Proceedings of the 4th international conference on Security and privacy in communication netowrks, 2008,
[34] K. Hoffman, D. Zage, and C. Nita-Rotaru. A survey of attack and defense techniques for reputation systems. Technical Report CSD TR #07-013, Purdue University, 2007
[35] K. Lai, M. Feldman, I. Stoica, J. Chuang, Incentives for cooperation in peer-to-peer networks, in: Workshop on Economics of Peer-to-Peer Systems, 2003.
[36] Girish Suryanarayana, Mamadou H. Diallo, Justin R. Erenkrantz, Richard N. Taylor, Architectural Support for Trust Models in Decentralized Applications, ICSE'06, 2006
[37] James Caverlee, Ling Liu, Steve Webb, SocialTrust: Tamper-Resilient Trust Establishment in Online Communities, JCDL'08, 2008.
[38] Selc¸uk, A., Uzun, E., Pariente, M.: A reputation-based trust management system for P2P networks. International Journal of Network Security **6**(3), 235–245 (2008)
[39] Azzedin, F., Ridha, A: "Feedback Behavior and Its Role in Trust Assessment for Peer-to-Peer Systems"; Accepted. To appear in Telecommunication Systems. 2009.