UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

THE REPUTATION, OPINION, CREDIBILITY AND QUALITY (ROCQ) SCHEME

Anurag Garg, Roberto Battiti

November 2004

Technical Report # DIT-04-104

The Reputation, Opinion, Credibility and Quality (ROCQ) Scheme*

Anurag Garg Roberto Battiti
Dipartimento di Informatica e Telecomunicazioni,
Università di Trento,
Via Sommarive 14, 38050 Povo (TN), Italy.
{garo,battiti}@dit.unitn.it

December 6, 2004

Abstract

An implicit assumption of trust in the participants is at the basis of most Peer-to-Peer (P2P) networks. However, in practice, not all participants are benign or cooperative. Identifying such peers is critical to the smooth and effective functioning of a P2P network. In this paper, we present the ROCQ mechanism, a reputation-based trust management system that computes the trustworthiness of peers on the basis of transaction-based feedback. The ROCQ model combines four parameters: Reputation (R) or a peer's global trust rating, Opinion (O) formed by a peer's first-hand interactions, Credibility (C) of a reporting peer and Quality (Q) or the confidence a reporting peer puts on the judgement it provides. We then present a distributed implementation of our scheme over FreePastry, a structured P2P network. Experimental results considering different models for malicious behavior indicate the contexts in which the ROCQ scheme performs better than existing schemes.

Keywords: Trust management, reputation, quality, credibility, autonomic systems, peer-to-peer systems.

1 Introduction

Trust is an important component of all human transactions and it becomes even more important when the transactions occur online. This is because of the high degree of anonymity and the absence of face-to-face interaction on the Internet. As a result, most human factors on which we base trust, such as manner, body language, intonations of

*This work was supported by the Provincia Autonoma di Trento through the WILMA Project.

speech, previous knowledge of a person's behavior etc. are absent. What then should be the basis of trust in such contexts? Several researchers and commercial organizations have been grappling with this issue in the past decade. One proposed solution has been to study the past interactions of a user and estimate his or her trustworthiness on this basis. This has led to a number of reputation-based trust management systems that allow a user to rate the transactions they have with other users. These ratings are then aggregated to form a global trust metric which is used to measure the trustworthiness of a user. How this aggregation should take place is a matter of contention and several competing schemes have been proposed.

The architectural and implementation details of the aggregation mechanism depend on the underlying network on which the online community is based. When the community is built on top of a traditional client-server network, a trusted third party exists which can be relied on to collect and aggregate opinions to form a global view. eBay feedback, Amazon customer review and the Slashdot distributed moderation systems are all examples where feedback from users is stored in a centralized trust database. The aggregation is performed in this centralized database and all users have access to the global reputations thus computed. When the community is built on top of a P2P network, the challenges of managing feedback become much harder. There is no centralized, trusted, reliable, always-on database and the collection, storage, aggregation and dispersal of trust information must be done in a distributed way. Relying on third parties for storage and dissemination makes the system vulnerable to tampering and falsification. The system must provide redundancy because peers may drop out of the network at any time. And the system must identify malicious peers that may collude with each other without access to the complete feedback information.

Hence, there are two separate but interrelated challenges that must be overcome by any distributed trust management system. The first is the actual computation of an appropriate trust metric that accurately reflects the trustworthiness of peers. The second is designing a system architecture that is robust against the challenges mentioned above.

With this in mind, we develop the ROCQ scheme (pronounced "rock") that computes peer Reputations (R) on the basis of Opinion (O), Credibility (C) and Quality (Q). First, we present the ROCQ model in Section 2 and show how our trust metric overcomes the limitations of previous metrics. In Section 3 we present a system architecture that builds on top of a structured P2P network to provide decentralized trust storage, aggregation and dissemination. We also discuss the design and implementation considerations for ROCQ. In Section 4 we describe a series of simulation-based experiments designed to evaluate the performance of ROCQ in a variety of settings. We conclude the paper with a discussion of ROCQ in the context of related work followed by a summary and discussion of future work.

2 The ROCQ Model

The ROCQ model uses three parameters (OCQ) for evaluating the reputation (R) of a peer. Hence ROCQ is an acronym for:
Reputation
Opinion
Credibility
Quality
We now discuss the individual components of ROCQ in greater detail.

2.1 Opinion

A peer forms an opinion about the amount of satisfaction it has derived from a transaction that it takes part in. Thus, we use O_{ij}^k to refer to peer i's opinion about its k^{th} transaction with peer j.
This opinion may then be shared with other peers in the form of feedback sent to the system. A peer may also choose to keep a record of its own first-hand experiences in the form of averaged opinions. As we shall see, a peer may choose to use its own stored opinion in addition to, or instead of, reputation information provided by the system.
Storing a record of every past transaction a peer has been involved in may be impractical due to scalability concerns. Instead, a peer may decide to only store the n most recent transactions or transactions that took place within a certain time limit. However, in system with a large number of peers and a large number of transactions even this may be impractical. Instead, peer can maintain an average opinion for each peer with which it has had an interaction.
O_{ij}^{avg} is peer i's estimate of the average amount of satisfaction it has received from peer j in past transactions. After each interaction with j, i updates O_{ij}^{avg}. Along with O_{ij}^{avg}, the peer i also stores the number of interactions, N_{ij}, it has had with j and the variance s_{ij}^2 in the behavior of j. Thus, the average local opinion is computed as follows:

$$O_{ij}^{avg} = \frac{\sum_k O_{ij}^k}{N_{ij}} \tag{1}$$

where O_{ij}^k is peer i's opinion of its k^{th} interaction with j. Each O_{ij}^k takes a value in $[0, 1]$ and represents peer i's satisfaction with peer j's behavior during the transaction.

2.2 Reputation

The reputation of a peer i is the end result of aggregating feedback about i from several other peers. It represents the global system-wide view of the average amount of satisfaction a peer is likely to derive through an interaction with i. The reputation of a peer is normalized so that it lies between 0 and 1 where a peer with reputation 0 will never behave satisfactorily and a peer with reputation 1 will always behave satisfactorily. In other contexts, reputation can take on other meanings as well. For instance, in a sensor

network it could be interpreted as the probability that a peer behaves correctly. In other cases, it could also be interpreted as the quality of service a peer is likely to provide. The reputation of peer i is computed by using feedback (in the form of opinions) from all peers that have interacted with i in the past. The opinions are weighted by the reporting peer credibility and the quality values they send with the their opinions.
The reputation of a peer j is computed at the peer m using reported opinions, the quality value sent by reporting peers and the credibility of reporting peers as follows:

$$R_{mj} = \frac{\sum_i O_{ij}^{avg} \cdot C_{mi} \cdot Q_{ij}}{\sum_i C_{mi} \cdot Q_{ij}} \tag{2}$$

where R_{mj} is the aggregated reputation of peer j, C_{mi} is the credibility of peer i according to peer m, O_{ij}^{avg} is the average opinion of j reported by i and Q_{ij} is the associated quality value reported by i. Thus peer m gives more weight to ratings that are considered to be of a high quality and that come from peers who are more credible in its eyes.
Depending on the implementation, a peer can send its averaged opinion O_{ij}^{avg} after each interaction or send only the satisfaction derived from the last transaction, O_{ij}^{k} where $k$transactions between i and j have taken place. Accordingly, the peer that receives this information can decide whether to store only the most recent value of O_{ij}^{avg} reported by each peer i or to store all opinions sent in a recent window. This window can be based on time or on the number of transactions.

2.3 Credibility

If a peer simply averaged the opinions it receives from other peers in order to form the reputation of other peers, it would open the system to a wide variety of attacks. There would be no way of distinguishing between true and false information. Peers could make false statements about other peers due to jealousy or to attract third parties to themselves. Peers could also collude with each other to falsely drive up their own reputations while lowering the reputations of other peers.
A simple form a credibility check was proposed by Aberer et al. [1] in their "complex" algorithm where the reputations of reporting peers were checked before their opinions were taken into account. Kamvar et al. [8] extended this by weighing the opinions of other peers with their reputations. We recognize credibility as a separate variable in the ROCQ algorithm, unrelated to a peer's reputation in the system as a peer may behave honestly in its transactions but behave maliciously when it rates the behavior of other peers.
In the ROCQ mechanism, the credibility of a peer is used to weigh the feedback it reports. If a peer gives wrong feedback about other peers its credibility rating is decreased and its subsequent reports have a reduced impact on the reputation of another peer. Similarly, if a peer's feedback is consistently good, i.e., in agreement with other reporting peers, its credibility rating goes up. Credibility ratings are based on first-hand experience only and unlike opinions are not shared with other peers. Credibility ratings are normalized so that they lie between 0 and 1.

C_{ij} or the credibility of peer j in the eyes of peer i is the confidence peer i has in peer j's opinions about other peers. All opinions expressed by peer j to peer i are weighed by this credibility value. C_{ij} is updated by peer i every time peer j reports an opinion. In addition, when peer i updates the credibility of peer j it uses the quality value furnished by peer j to decide the amount of modification. This is because a peer should not be penalized for an incorrect opinion that is based on a small number of interactions and/or interaction that led to variable satisfaction where this is explicitly stated by the reporting peer through a low quality rating.

When a peer reports an opinion to another peer for the first time, its credibility is set to 0.5. Thereafter, every time it reports an opinion to this peer, its credibility is adjusted according to the following formula:

$$C_{mi}^{k+1} = \begin{cases} C_{mi}^{k} + \frac{(1-C_{mi}^{k})\cdot Q_{ij}}{2} & \text{if } |R_{mj} - O_{ij}^{avg}| < s_{mj} \\ C_{mi}^{k} - \frac{C_{mi}^{k}\cdot Q_{ij}}{2} & \text{if } |R_{mj} - O_{ij}^{avg}| > s_{mj} \end{cases} \tag{3}$$

where C_{mi}^{k} is the credibility of peer i in the eyes of peer m after k reports to peer m, O_{ij}^{avg} is the opinion being currently reported by peer i, Q_{ij} is the associated quality value, R_{mj} is the aggregated reputation value that peer m computed for j and s_{mj} is the standard deviation of all the reported opinions about peer j. Thus, credibility updates take the reported quality value into account. Because an opinion with a smaller quality value does not count as much, the change in credibility is proportionately lower. At the highest reported quality value of 1, a reported rating that falls within one standard deviation of the aggregated reputation, increments the credibility of the reporting peer by half the amount required for credibility to reach 1. A reported rating outside this region results in the credibility rating dropping to half the previous value.

In this way, if a reporting peer is malicious, its credibility rating is gradually reduced since its opinion does not match that of other peers. And a peer with a lower credibility value therefore contributes less to the aggregated reputation.

2.4 Quality

Unlike existing trust management systems, the ROCQ scheme allows a peer to determine the confidence of its feedback. This serves two important purposes. First, giving incorrect feedback can decrease the credibility of a peer. Therefore, a peer can lower the quality value for opinions about which it is not very sure, therefore risking less loss of its credibility in case its judgment is incorrect. Not all transactions are created equal and it may be harder to judge the honesty of a transaction partner in some transactions. Second, transactions vary in importance. A peer could behave honestly in several small transactions and build a high reputation and then cheat in big transactions. If all transactions are weighed equally, a peer can simply oscillate between building up a reputation in small transactions followed by cheating in a few large transactions. Finally, in a system where peers may dynamically alter their behavior, a peer is likely to be much more confident about its opinion about another peer if it has had a large number of interactions with that peer and those

interactions have always yielded roughly the same amount of satisfaction. Like the other parameters, quality is also normalized to lie between 0 and 1.

When peer i sends its updated average opinion about j (O_{ij}^{avg}) to other peers, it also sends an associated quality value, Q_{ij}. Q_{ij} represents the quality peer i attaches to the opinion information it is sending. Q_{ij} enables a peer to express the strength of its opinion. Its value can depend on the context of the interaction as well as on the past transaction history. In our current implementation, quality values are computed solely on the basis of the number of interactions and the variance in the opinion.

We assume that j's trust behavior is a normally distributed random variable. Through interactions with j, peer i makes observations of this random variable resulting in a sample. The sample mean and standard deviation are then simply O_{ij}^{avg} and s_{ij}.

The quality value of the opinion (Q_{ij}) is defined as the confidence level that the actual mean trust rating for a peer lies within the confidence interval:

$$O_{ij}^{avg} \cdot (1 \pm \frac{r}{100}) \tag{4}$$

where r is a system parameter that denotes the size of the confidence interval as a percentage of the sample mean. We ran experiments with r values ranging from 5 to 30 and found that a confidence interval of 10% of the sample mean (and thus $r = 10$) resulted in the best performance. Using too high a value for r produced useless quality values, as large variations in peer behavior were allowed without any decrease in the quality. Similarly, too low a value of r resulted in excessively low quality values.

Since the actual mean and standard deviation are unknown, we use the *Student's t-distribution* to compute the confidence levels. Note that the usual idiom is inverted here in that we specify the interval and wish to compute the probability that the actual mean lies within the interval as opposed to normal practice where confidence level is known and the interval is computed.

The t-value for the *Student's t-distribution* is given by the following equation:

$$t = \frac{r}{100} \cdot \frac{O_{ij}^{avg} \cdot \sqrt{N_{ij}}}{s_{ij}} \tag{5}$$

And the quality value is computed as:

$$Q_{ij} = 1 - B\left(\frac{(N_{ij}-1)}{(N_{ij}-1)+t^2}; \frac{1}{2} \cdot (N_{ij}-1), \frac{1}{2}\right) \tag{6}$$

where B is the *Incomplete Beta Function* defined as $B(z;a,b) \equiv \int_0^z u^{a-1} \cdot (1-u)^{b-1}\ du$

Thus an opinion is of greater quality when the number of observations on which it is based is larger and when the interactions have been consistent, resulting in a smaller variance. When the number of observations is high but they do not agree with each other, the quality value is lower.

When a peer has had only one interaction with the other peer, equations 5 and 6 cannot be used since sample variance is undefined for a sample size of one. Instead, a default quality

value of 1 is used in this case. If a lower value, such as 0.5, is used, the opinions sent by malicious peers during the initial interactions, when there is little credibility information, would overwhelm the opinions of the good peers, as malicious peers would always report a quality value of 1 for their opinions.

3 System Architecture

While the ROCQ trust model is independent of the underlying architecture, its effectiveness clearly depends on the system architecture and the ROCQ implementation. Since there is no centralized database, any implementation needs to collect, store and disseminate reputation information in a distributed way. Moreover, this should be done in a scalable, efficient and cost-effective manner.
Our implementation of ROCQ assumes a structured overlay network that provides a secure, deterministic and reliable way to route messages. Examples of such networks include Chord, CAN and Pastry. These networks use Distributed Hash Tables (DHT) to map objects to a keyspace. Nodes in the network are then responsible for certain ranges of the keyspace. DHTs provide the functionality required by our implementation of ROCQ to use **score managers**. A score manager for a peer is another peer in the network that stores all trust information related to that peer. All feedback pertaining to that peer and requests for the reputation of that peer are routed to the score manager. Hence, score managers function as the decentralized trust database.

3.1 Score Managers

The underlying DHT overlay structure randomly and uniformly designates M score managers for each peer in the network. This is done by hashing a well-known identifier for a peer, such as its IP address. The peer closest to the resulting hash ID in the key space is assigned as a score manager for that peer. Multiple score managers are assigned either by using multiple hash functions or by assigning the M peers that are closest to the hash ID in the key space. Since the score managers are selected from peers within the network itself, each score manager is responsible for M peers on average. This allows any entity that knows the identifier of a peer to contact its score managers. Multiple score managers provide the reputation system with redundancy so that the failure of a few score managers does not affect the trust management system. Since the score managers are also peers in the network, we use the term "peer" to refer exclusively to a peer when it is not acting in its capacity as a score manager.
After each interaction, both transacting peers report their updated opinions along with the associated quality values to the score managers responsible for their counterpart. The score managers aggregate these opinion values they receive for the peers to construct their global trust value or *reputation*. The score managers therefore represent the global, system-wide view of a given peer's behavior, updated every time a new opinion is received.

Score managers also respond to reputation query requests by peers wishing to transact with a peer for which they are responsible. The requesting peer receives multiple responses along with the associated quality values which are then aggregated a second time by using the credibility of the score managers themselves to compute the final reputation value for the prospective transaction partner.

Along with the aggregate reputation R_{mj} for a peer j, a score manager m also stores the number of opinions it has received about j, N_{mj} and the variance s^2_{mj} in the reported opinions. This information is used to compute the quality value that the score manager attaches to a reputation value. The computation of the quality of reputation values at a score manager is similar to that of the individual peers' computation of quality values for their averaged opinions described in equations 5 and 6. So, the quality of a reputation value is simply the confidence level that the actual mean reputation of the peer is within $r\%$ of the sample mean.

If the reputation value of a peer at the score manager has been calculated by using the opinion of a single voter only, a quality value of 1 is returned. The reason for this is the same as described in the previous section.

3.2 Retrieval of Trust Information

When a peer wants to know the reputation of another peer before interacting with it, it locates the M score managers for the peer by using the DHT substrate and asks them for the reputation of the peer in question. Each score manager responds with a reputation value and an associated quality value. The peer then computes the average reputation for the peer in question, R^{avg}_{ij} by using the quality values and the credibility values of the score managers since a score manager itself may also be malicious and send the wrong reputation values. In this way, multiple score managers allow the system to cope with malicious behavior.

In the case of reputation retrieval, a peer aggregates the responses from the reporting score managers by using the reported quality value and the stored credibility value of the reporting score managers. The aggregation is performed in exactly the same way as shown in equation 2 except that the reputation values R_{mj} are aggregated instead of the opinions O^{avg}_{ij}.

$$R^{avg}_{ij} = \frac{\sum_y R_{yj} \cdot C_{iy} \cdot Q_{yj}}{\sum_y C_{iy} \cdot Q_{yj}} \tag{7}$$

where R_{yj} is a reputation value received from a score manager y about peer j.

We should mention that while reputation is aggregated by using a weighted average of the reported reputations in ROCQ, it is possible to implement other strategies as well. For instance, score managers may decide to ignore opinions with associated quality values below a certain threshold or from peers with credibility below a certain threshold.

The final average reputation value – formed by two aggregations, first at the score managers and second at the requesting peer – is then used to decide whether the transaction should proceed. If a peer has had interactions with the prospective partner before, it may have already formed an opinion value for this peer. In this case, the peer may wish to

prefer its own first-hand experience to the information being provided by the trust management system. ROCQ recognizes this and in Section 4.2 we present results that evaluate the validity of this approach.

3.3 Resource Requirements

Since each peer has M score managers associated with it, a transaction between two peers results in $2M$ messages to the score managers. Similarly, a trust query from a peer to the score managers of its potential transaction partner results in M messages to the score managers and M responses. As the number of score managers M does not depend on the number of peers in the network, the network traffic increases by a constant factor due to the ROCQ scheme.
Assume that a peer transacts with K peers on average. In the hypothesis of random and uniform selection of score managers, on average each peer acts as a score manager for M peers and stores the last reported opinion from each of the K transaction partners for these M peers. The average storage requirements for being a score manager are therefore $O(KM)$. Each peer also stores its own average opinion for the K peers it has interacted with, resulting in $O(K)$ storage. Hence the total average storage requirements for a peer are $O(KM)$.
Each transaction (preceded by two trust queries) results in reputation and quality computations at $2M$ score managers (M for each transacting peer), followed by weighted averages of the reputation being computed at each transacting peer, followed by a single average opinion update at each of the $2M$ score managers.

4 Experimental Results

We now present the results of our simulation-based experiments with ROCQ. Results from initial experiments that studied the feasibility and effectiveness of ROCQ were presented in [7]. The results we present below are more detailed and present several enhancements to the algorithm and the implementation that have since been made.

4.1 Methodology

For our experiments, we use FreePastry [12], an open-source implementation of Pastry that is written in Java. We use FreePastry to create a virtual P2P network and to deliver messages between peers. In FreePastry, messages can be delivered directly to a specific peer (if the exact ID is known) or they can be routed through the network (the message is received by the peer with the smallest ID greater than the key specified in the message).
ROCQ is implemented as an application that runs on top of individual Pastry nodes. We now describe the simulation setup including default parameter settings, network topology, decision metrics etc.

Number of Peers and Interactions. Unless stated otherwise, each experiment simulates a network with 200 peers where 50000 transactions take place (i.e., each peer has an average of 250 interactions). Both participants of each interaction are chosen randomly. The default number of score managers storing reputation ratings for each peer is 6 unless specified otherwise. Each experiment was performed 10 times and the resulting average of the results is plotted, along with a confidence interval of size $\frac{s}{\sqrt{10}}$ where s is the standard deviation.

Performance Metric. The performance of our scheme is evaluated as the number of correct decisions made by honest nodes (i.e., interactions with good peers that went ahead plus interactions with malicious peers that were avoided) as a proportion of the total number of decisions made by honest nodes. Because we are interested in the steady state performance of the system, the initial interactions that take place when no information about the reputation of a peer can be found are not counted. Moreover, we are not interested in decisions made by dishonest nodes, hence they are not counted in the masurement of performance.

Type of Maliciousness. We simulate two different kinds of maliciousness. A peer can be malicious in the base system, i.e., behave maliciously when interacting with other peers and/or it may be malicious in the reputation system. In the former case, two good peers (and two bad peers) give each other a rating of 1 after interacting whereas if a good and a malicious peer interact they both give each other a rating opinion of 0. In the latter case, the peer behaves maliciously in its capacity as a score manager and sends incorrect reputation values to requesting peers. The reputation values thus sent are $1 - R$ where R is the actual reputation value. The reason we chose this model of maliciousness as opposed to say sending an arbitrary number, is that this is the worst possible form of maliciousness. A peer may act maliciously in either the base or the reputation system alone or in both systems.

Decision Metric. When a peer has had several interactions with another peer it may choose to rely on its own first-hand experience rather than on the information given to it by the trust management system. It may also choose to rely on a combination of the two. Hence the basis for deciding whether to proceed with an interaction can be:

- **Reputation:** The source relies solely on the aggregate of reputation values retrieved from the score managers.
- **Local Opinion:** The source relies on its own averaged local opinion of the target if it has had at least 5 interactions with it, otherwise Reputation is used.
- **Reputation plus Local Opinion (default):** If the source has had at least one previous interaction with the target, the Reputation and Local Opinion values are averaged. Otherwise, only the Reputation is used.

When there is no information available to determine the level of trust of a peer the interaction takes place. However, these interactions are counted as initial transactions and are not used for measuring the performance of the ROCQ mechanism.

Interaction Threshold. Once the target's trust metric has been calculated, the source decides whether to proceed with the interaction by using one of the two following decision algorithms.

- **Deterministic Selection (default):** The interaction proceeds if and only if the trust metric is greater than or equal to 0.5.
- **Probabilistic Selection:** The probability that the interaction will proceed is equal to the trust metric value. In this case, there is a non-zero probability that a source will interact with a target that has a very low trust rating as long as the rating is not 0.

Network topology. Each interaction involves a source peer that initiates the transaction and the target peer that responds. The source of the interaction is always chosen randomly from all the peers of the system, while the target is selected according to the *Network Topology*:

- **Random (default):** The target is selected from all the peers in the network.
- **Power Law:** In this topology, the probability of a peer being selected as a target varies according to a power law distribution. Such network topologies are an accurate representation of the Internet [6].
- **Tribes:** The peers are divided in groups of k peers and the target is randomly selected from the peers in the same group as the source.
- **Overlapped:** The target is selected randomly from the $k/2$ peers that directly precede and the $(k/2) - 1$ peers that directly follow the source peer in the peer array.

In the last two topologies, the target can be randomly selected from the population as a whole with a probability of 10%.

4.2 Choosing the Right Decision Metric

In this set of experiments we evaluate the impact of the different decision metrics for deciding the trustworthiness of a peer. We ran the experiments with peers behaving maliciously in just the base system and in both the base and reputation systems. The results are shown in Fig. 1 and Fig. 2.
In both experiments we vary the percentage of malicious peers in the network from 5% to 90%. Here, malicious peers act maliciously in a consistent fashion. In [7] we showed some preliminary results that compared when peers acted maliciously only some of the time. There are several things of note in Fig. 1. First, we see that for all three decision

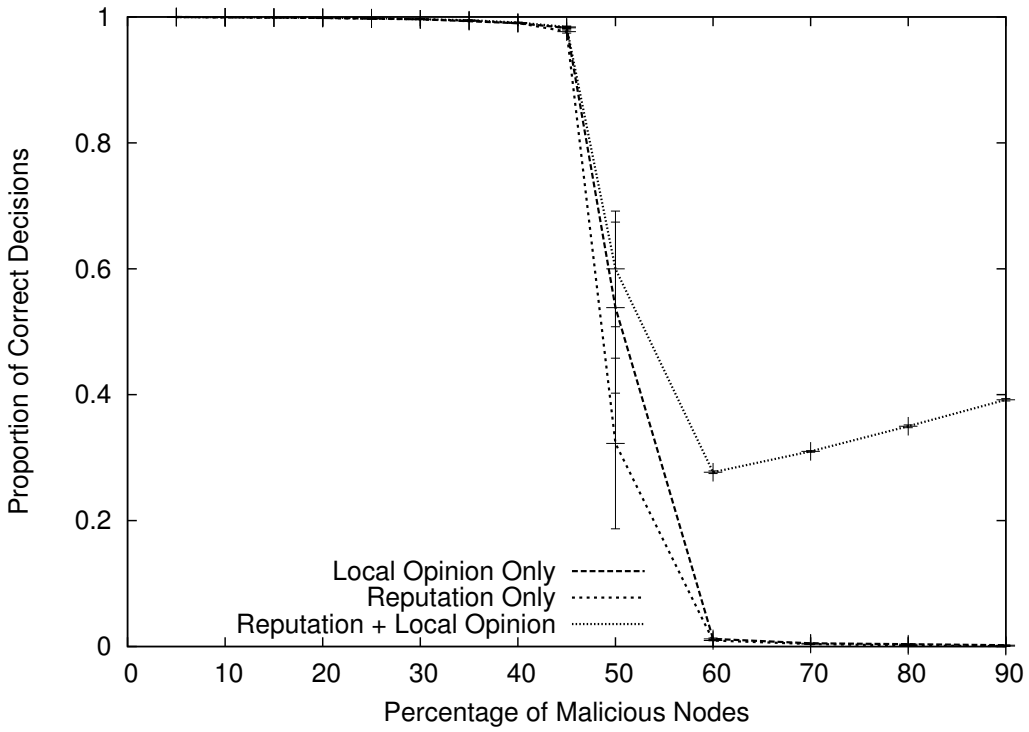

Figure 1: Performance of ROCQ using Reputation, Local Opinion and Both with Maliciousness in the Base System Only.

metrics the proportion of correct decisions falls steeply when the percentage of malicious nodes exceeds 50%. This is because when the majority of nodes in the network are malicious the dominant ethic of the system becomes that of the malicious peers and non-malicious peers are mistakenly labelled as untrustworthy. Incorrect feedback from malicious peers overrides the honest feedback being provided by the non-malicious peers. Second, we see that the combination of reputation and local opinion significantly outperforms both the other strategy when the percentage of malicious nodes in greater than 50%. When the proportion of malicious nodes rises above half, the information being furnished by the reputation system becomes faulty and non-malicous nodes are more likely to make the correct decision when they rely on their own first-hand experience.

In Fig. 2 the superiority of the combined scheme is again evident. But we see another interesting feature. The performance of our scheme decreases till the percentage of malicious nodes is about 50% and then rises again. As the percentage of malicious nodes increases, a larger proportion of interactions take place between two malicious nodes. These nodes give each other an opinion rating of 1 but when this opinion is reported to the score manager – which itself has a high likelihood of being malicious – it inverts this rating and the reputation of the malicious nodes is correctly reduced to 0. Therefore, a large number of interactions with malicious nodes are avoided, thus improving the per-

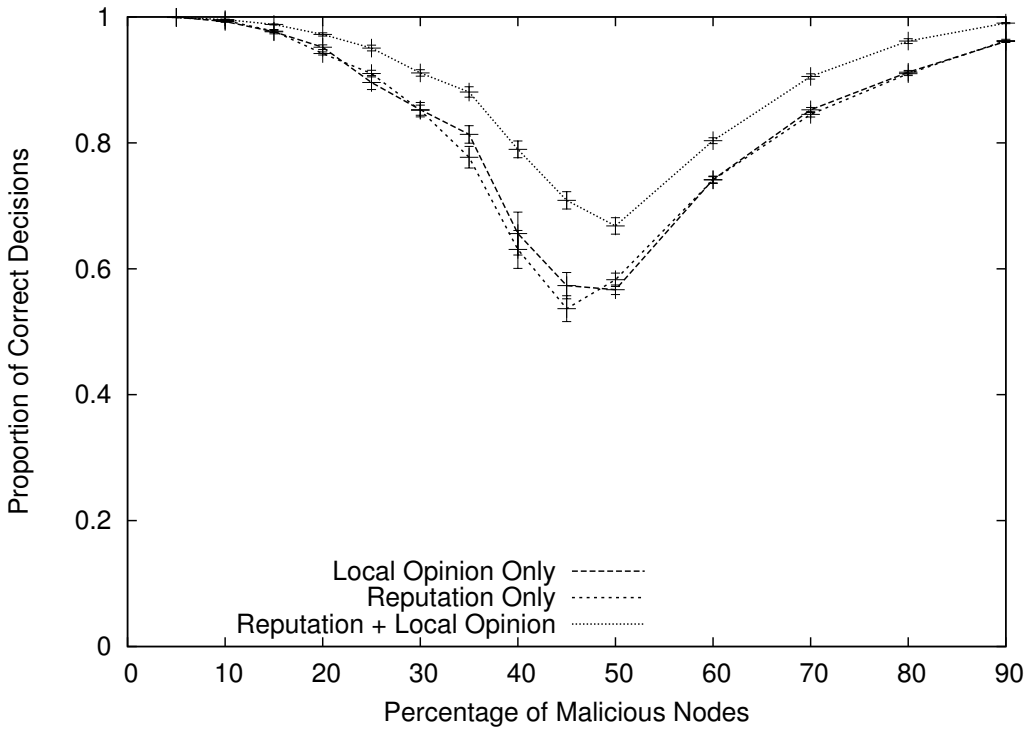

Figure 2: Performance of ROCQ using Reputation, Local Opinion and Both with Maliciousness in Both the Base and Reputation Systems.

formance of our system.

4.3 Comparison with the Aberer-Despotovic Scheme

In this experiment we compare the performance of the ROCQ scheme with the trust management scheme proposed by Aberer and Despotovic in [1]. We implemented their scheme in Pastry and ran experiments with the same number of nodes and interactions as in our scheme (200 and 50000 respectively). Aberer and Despotovic proposed two schemes, the *simple* and the *complex*. In the simple scheme, nodes give equal weight to all reporting nodes. In the complex scheme, only reports from nodes that exceed a given trust threshold are taken into account.

In the experiments conducted by Aberer and Despotovic, there were two distinct phases. In the first phase, a number of interactions were performed allowing nodes to build up trust data. In the second phase, their trust management scheme decides whether a given interaction can take place or not. However, unlike the simulations they performed, we did not start making trust assessments after the first interaction period had ended. Instead we have only one phase and interaction decisions are made before each potential interaction on the basis of trust data that has been accumulated so far.

We feel this model is closer to reality as nodes would want to know the nature of a node

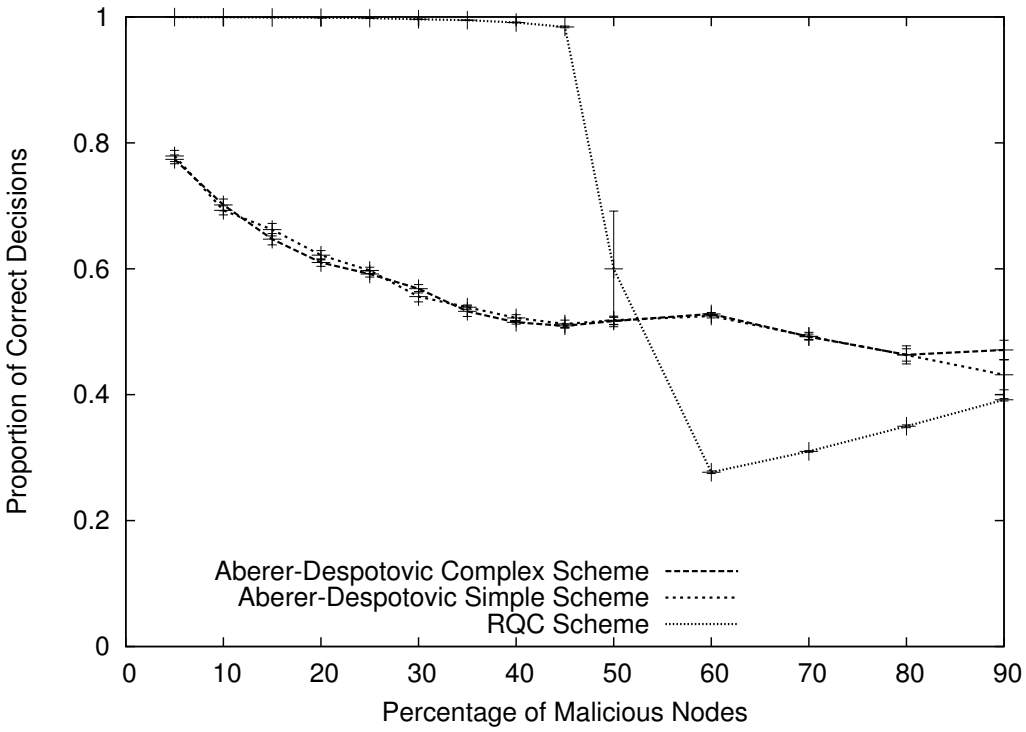

Figure 3: Comparison of the RQC Scheme with the Aberer-Despotovic Scheme

before interacting with it instead of waiting for a large number of interactions to finish. As we can see in Fig. 3 and 4, our scheme performs consistently better than the Aberer-Despotovic scheme in terms of the proportion of correct decisions made.
Figure 3 compares the performance of the two schemes when there is maliciousness in the base system only whereas Fig. 4 compares the performance when there is maliciousness in both the base and the reputation system. In both cases, ROCQ outperforms the Aberer-Despotovic scheme.

4.4 Deterministic vs. Probabilistic Target Selection

This experiment shows how the performance of our mechanism is influenced by changing the way in which a source peer determines if it wants to interact with a particular target. In this and all subsequent experiments a combination of reputation and local opinion is used as our decision metric because this metric outperforms the other two metrics of reputation only and local opinion only.
In figure 5 and 6 we compare the probabilistic and the deterministic target selection methods with the maliciousness in the base system and in both base and reputation systems.
In both cases, the ROCQ performs better when deterministic target selection is used. With deterministic selection, once a peer's reputation value goes below 0.5 that peer will never be selected as for interaction by a non-malicious node again. On the other hand, when

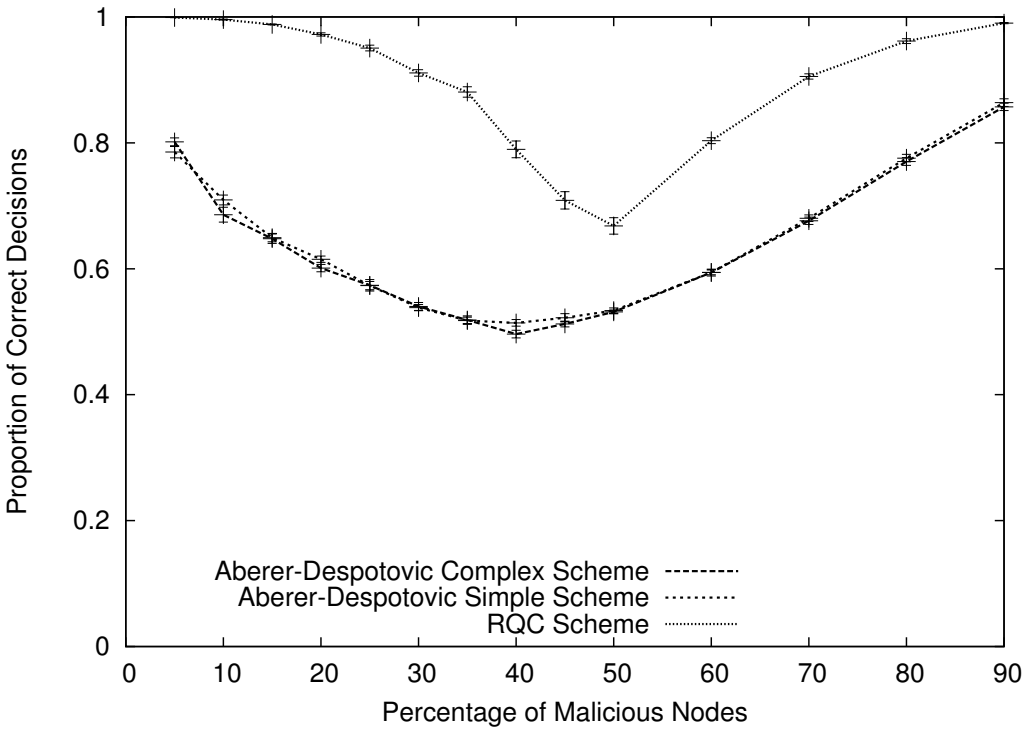

Figure 4: Comparison of the RQC Scheme with the Aberer-Despotovic Scheme

peers with a trust metric of below 0.5 are sometimes chosen for interaction, they are quite likely to be malicious.
The objective of this experiment is to test if a node that is wrongly labelled as malicious can benefit from the probabilistic target selection and improve its reputation value through subsequent interactions. As our results show, any such gain is more than nullified by the loss through the additional interactions with malicious peers.

4.5 Impact of Network Topologies

In this set of experiments we examine the influence network topology has on the detection of malicious peers. We test all four topologies that were described above in Section 4.1. The results are shown in Fig 7 and Fig 8.
The ROCQ mechanism performs best when the network topology is "Overlapping" or "Tribes". This is because with these topologies, each peer has a very small number of regular transaction partners (19 since the tribe size is 20). Each peer interacts with peers outside this group only 10% of the time. As a result, each peer has the opportunity form a local opinion through first-hand interactions for all of these partners. Therefore, even if the majority of peers in the network are malicious, a good peer can rely on its local opinion to take the correct decision. With the "Power Law" and "Random" topologies nodes do not interact with their transaction partners as frequently since the number of

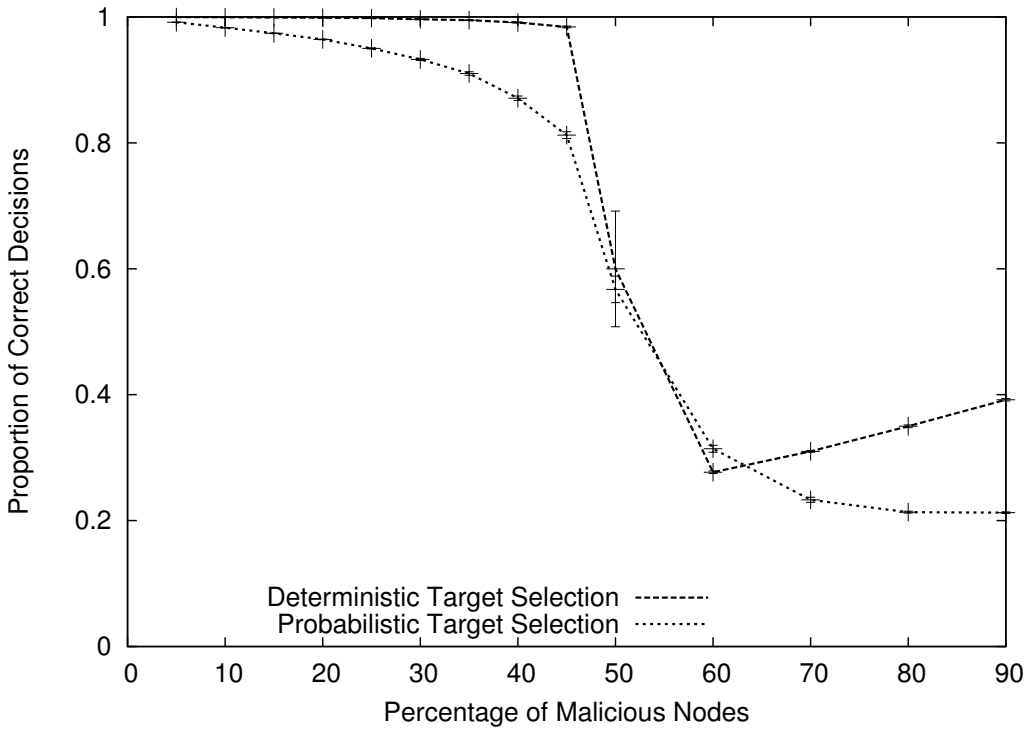

Figure 5: Deterministic and Probabilistic Target Selection With Maliciousness in the Base System Only.

partners is much larger.

4.6 The Impact of Churn in the Network

So far we have only considered the case when the network is reliable and all the nodes are alive all of the time. However, such a scenario is unrealistic and in real P2P networks nodes go up and down all the time in a phenomenon known as **churn**. There has been some research on how well different DHT implementations cope with churn [11]. In this set of experiments, we examine what effect churn has on a reputation system. In these experiments, our definition of churn is somewhat limited in that no new nodes join the network. Instead, churn is implemented through each node having a fixed probability of going down during every round of interactions. The more complex case of nodes joining and leaving the network is under examination and the results will be published in the future.

In this set of experiments we examine the scenarios when 0%, 10% and 20% of the nodes are down at any given time. In these experiments, we fix the percentage of malicious nodes at 30% and these nodes consistently act maliciously. As in the previous experiments, we use the reputation plus local opinion system to compute the trust value of a node and we examine the cases when nodes are malicious in the base system only and in

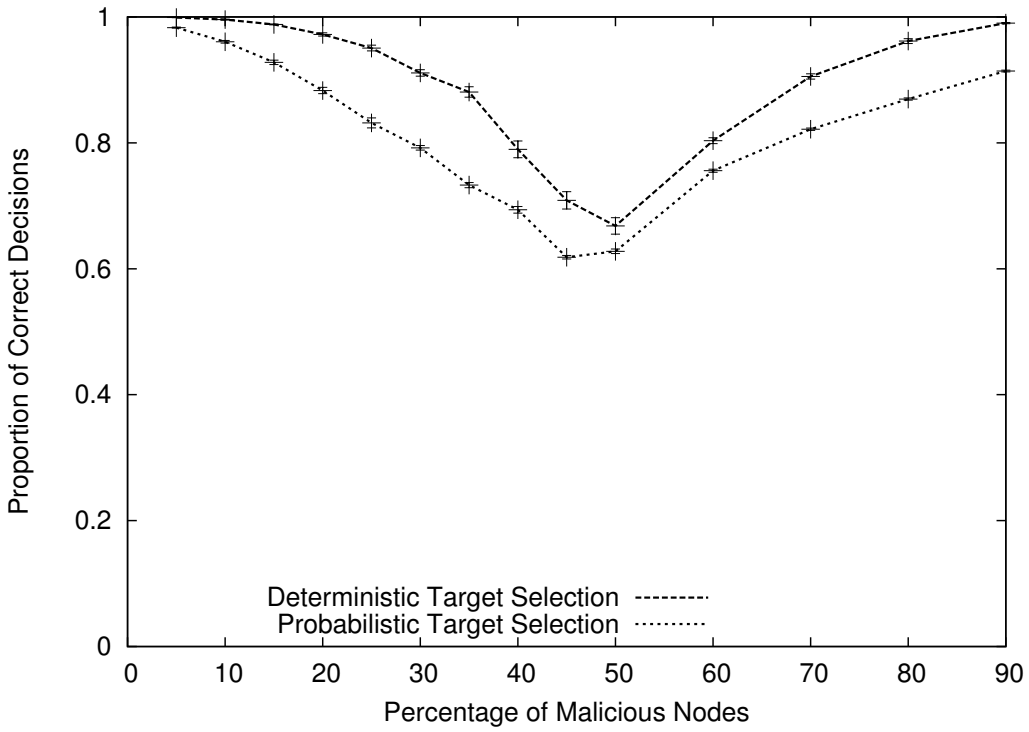

Figure 6: Deterministic and Probabilistic Target Selection With Maliciousness in Both Base and Reputation Systems.

both the base and reputation systems (i.e., both as ordinary peers and as score managers). We repeat each set of experiments with 3, 6 and 10 score managers per each peer in the network. This is because a higher number of score managers adds redundancy to the network. The impact is evident in Fig. 9 and in Fig. 10 where an increase in the percentage of dead nodes decreases the proportion of correct decisions but additional score managers alleviate this decrease to some degree.
However, what is more remarkable is that even with 20% of the nodes inoperational at any time, the decrease in the proportion of currect decisions is not as dramatic and the reputation system keeps functioning. In both cases, the decrease in the proportion of correct decisions is about $10 - 15\%$. Hence, the ROCQ scheme is robust against node failure and can be made mroe robust by increasing the number of score managers per node.

5 Related Work

Initial efforts at trust management in electronic communities were based on centralized trust databases. The eBay rating system used for choosing trading partners where each participant in a transaction can vote $(-1, 0, 1)$ on their counterpart, the Amazon customer

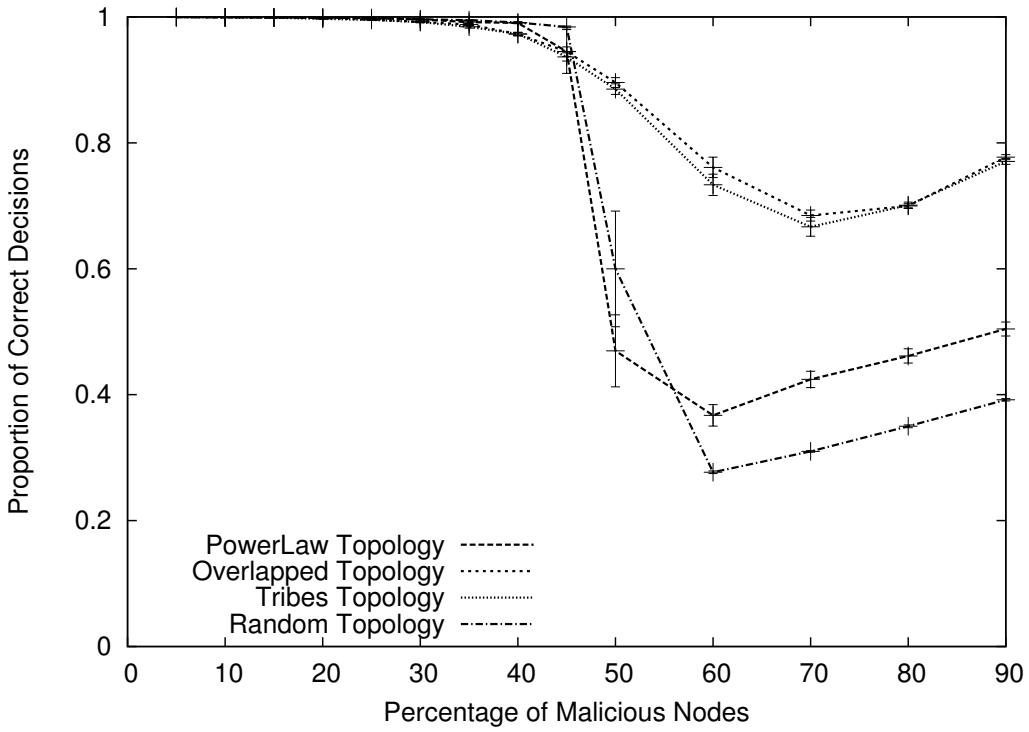

Figure 7: Impact of Different Network Topologies With Maliciousness in the Base System.

review system and the Slashdot self-moderation of posts [9] are all systems where the ratings are provided by peers but are stored in a central database. Many such reputation systems have been studied in the context of online communities and marketplaces [5, 10, 15].

Aberer and Despotovic introduced a scheme [1] using a decentralized storage system P-Grid to store and retrieve trust information. Peers file complaints against other peers who they feel have behaved malicioulsy and the complaints are stored at other peers called agents. Two algorithms are described to compute the trustworthiness. The first relies on a simple majority of the reporting agents' decisions and the second checks the trustworthiness of the reporting agents themselves and disallows any reports coming from untrustworthy agents.

Cornelli et al. [3, 4] propose a mechanism built on the Gnutella network, where a peer uses a secure polling mechanism to ask its neighbors about interactions they may have had with a specific peer to gauge its trustworthiness. The scope of the messages querying trust is limited by the Gnutella architecture design. Their work is directed at file-sharing networks and the objective is to find the most trusted peer that possesses a given resource.

Kamvar et al. [8] use a different approach and assume that trust is transitive. Therefore, a peer weighs the trust ratings it receives from other peers by the trust it places in the reporting peers themselves. Global trust values are then computed in a distributed fashion

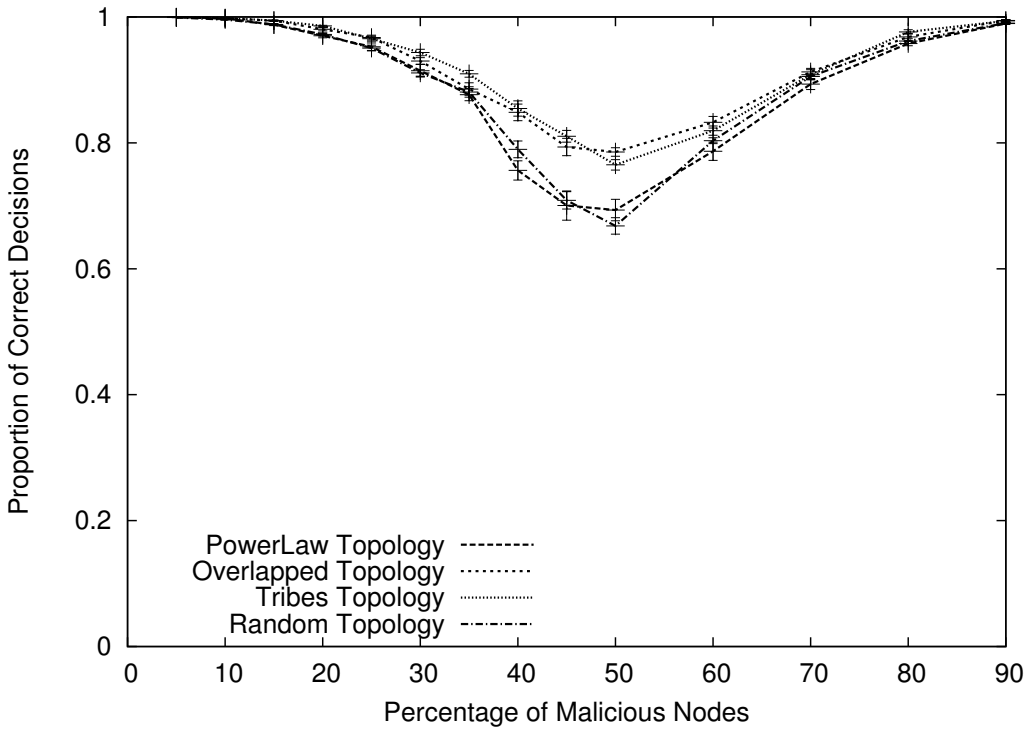

Figure 8: Impact of Different Network Topologies With Maliciousness in Both Base and Reputation Systems.

by using a trust matrix at each peer. Trust values asymptotically approach the eigenvalue of the trust matrix, conditional on the presence of pre-trusted peers that are always trusted. Buchegger et al. [2] propose a modified Bayesian approach to trust. Like Damiani et al. they separate a peer's reputation (performance in the base system such as file-sharing, routing, distributed computing etc.) and its credibility (performance in the reputation system). Xiong et al. [14] have recently proposed a more complex model called PeerTrust where they include peer trustworthiness, credibility and transaction context. Sun et al [13] have proposed an incentive based scheme that works on unstructured P2P networks. However, their mechanism is limited to answering queries in file-sharing networks.

6 Conclusions

In this paper we have proposed the ROCQ mechanism that creates a reputation-based trust framework that deals with various kinds of maliciousness. Our implementation of ROCQ on FreePastry produces excellent results and makes the correct decision almost 100% of the time as long as malicious nodes are not the majority in the network. If malicious nodes are a majority, ROCQ still returns the correct decision $25 - 30\%$ of the time if the system-wide reputation values are combined with local information. This exceeds the

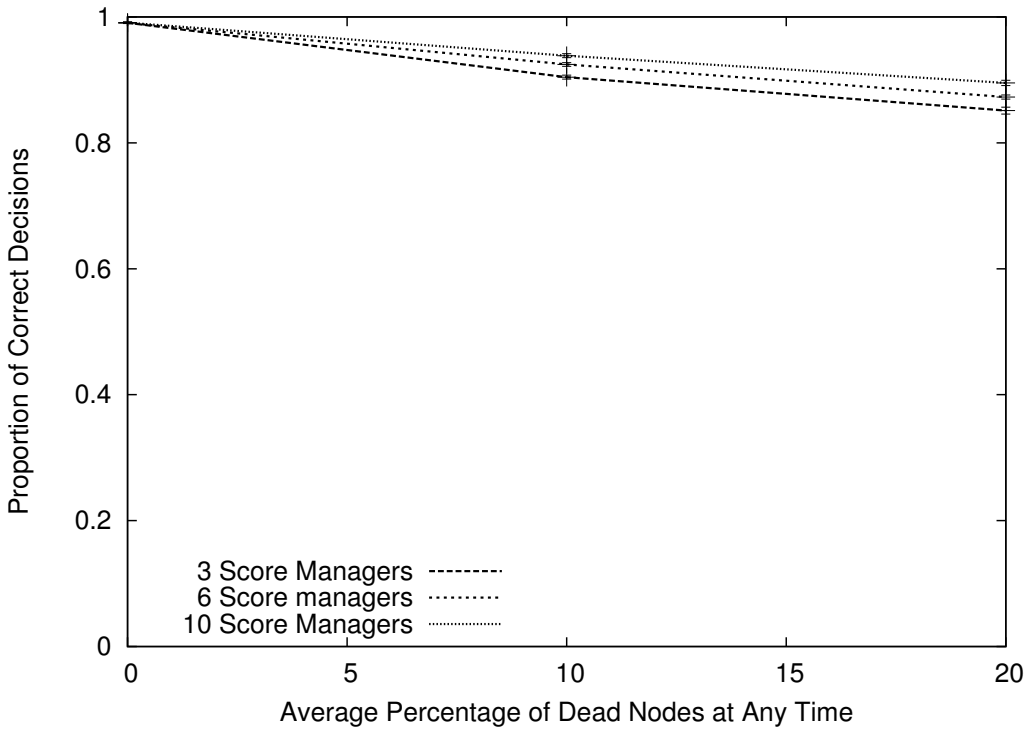

Figure 9: The Effect of Churn on ROCQ with Maliciousness in the Base System Only

performance of other reputation-based trust mechanisms in similar settings. Finally, we demonstrate that ROCQ is resilient to node failure and this tolerance can be increased by increasing the number of score managers per peer.

7 Acknowledgments

We would like to thank Gianni Costanzi for his valuable contribution in developing the simulation software.

References

[1] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *CIKM*, pages 310–317, 2001.

[2] S. Buchegger and J.-Y. L. Boudec. A robust reputation system for p2p and mobile ad-hoc networks. In *Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[3] F. Cornelli, E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a p2p network. In *Eleventh International World Wide Web Conference*, Honolulu, Hawaii, May 2002.

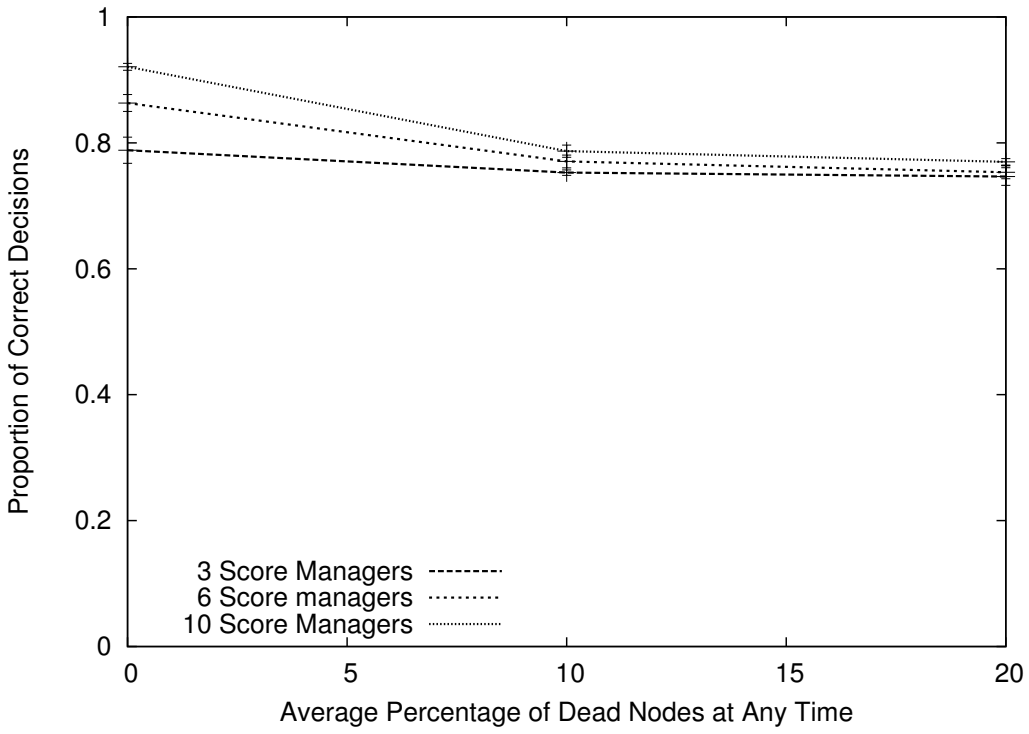

Figure 10: The Effect of Churn on ROCQ with Maliciousness in Both Base and Reputation Systems.

[4] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati. Managing and sharing servents' reputations in p2p systems. *IEEE Transactions on Data and Knowledge Engineering*, 15(4):840–854, July/August 2003.

[5] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 150–157. ACM Press, 2000.

[6] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *SIGCOMM*, pages 251–262, 1999.

[7] A. Garg, R. Battiti, and G. Costanzi. Dynamic self-management of autonomic systems: The reputation, quality and credibility (rqc) scheme. In *To appear in The 1st IFIP TC6 WG6.6 International Workshop on Autonomic Communication (WAC 2004)*, Oct. 2004.

[8] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the twelfth international conference on World Wide Web*, pages 640–651. ACM Press, 2003.

[9] C. Lampe and P. Resnick. Slash(dot) and burn: distributed moderation in a large online conversation space. In *Proceedings of the 2004 conference on Human factors in computing systems*, pages 543–550. ACM Press, 2004.

[10] P. Resnick, R. Zeckhauser, J. Swanson, and K. Lockwood. The value of reputation on ebay: A controlled experiment, June 2002.

[11] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *In Proceedings of USENIX Annual Technical Conference*, 2004.

[12] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, Nov. 2001.

[13] Q. Sun and H. Garcia-Molina. SLIC: A Selfish Link-based Incentive Mechanism for Unstructured Peer-to-Peer Networks. In *24th International Conference on Distributed Computing Systems (ICDCS'04), Tokyo, Japan*, Mar. 2004.

[14] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust in peer-to-peer communities. *IEEE Transactions on Data and Knowledge Engineering, Special Issue on Peer-to-Peer Based Data Management*, 16(7):843–857, July 2004.

[15] G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *Proceedings of the Thirty-second Annual Hawaii International Conference on System Sciences-Volume 8*, page 8026. IEEE Computer Society, 1999.