Incentives for Combatting Freeriding on P2P Networks.

Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina

Stanford University

Abstract. We address the freerider problem on P2P networks. We first propose a specific participation metric, which we call a peer's EigenTrust score. We show that EigenTrust scores accurately capture several different participation criteria. We then propose an incentive scheme that may be used in conjunction with any numerical participation metric. We show that, when these incentives are used in conjunction with EigenTrust scores, they reward participatory peers but don't exclude less active peers.

1 Introduction

A notable problem with many of today's P2P file-sharing networks is the abundance of *freeriders* on the network – peers who take advantage of the network without contributing to it. Up to 70% of Gnutella clients do not share any files, and nearly 50% of all responses are returned by 1% of the peers [1]. This abundance of freeriders, and the load imbalance it creates, punishes those peers who do actively contribute to the network by forcing them to overuse their resources (e.g. bandwidth).

We address this problem by providing incentives for peers to make active contributions to the network. For example, active participators may get preference when they are competing for another peer's resources, such as bandwidth. Our approach is simple: each peer gets a certain participation score, and it receives rewards based on its participation score. The challenges here lie in how to quantify participation in a manner that is both accurate and robust to malicious peers, how to store and manage these scores in a secure and distributed fashion, and how to structure incentives that reward active participators without completely excluding peers that are less active.

Previous work in this area has focused primarily on currency-based systems wherein peers gain currency for uploading files, and use currency when downloading files [6, 4]. We take a different approach, rewarding peers with high participation scores with advanced services, such as faster download times or an increased view of the network. Our approach may be used in conjunction with currency-based approaches.

In this work, we describe a scoring system that accurately quantifies participation, even in the presence of malicious peers trying to subvert the system, and we propose some incentives, and show empirically that these incentives benefit participatory peers in a fair manner.

2 Assessing Participation Levels

We believe that the level of participation of peer i should be based on two criteria.

1. The number of authentic uploads peer i provides to the network. Clearly, in a file-sharing network, the more authentic uploads peer i provides to the network, the greater the service does for the network.

2. The participation level of the peers whom peer i services. If a peer services freeriders, it is doing less for the network than a peer who services peers who will share the files they download.

This second criteria has a recursive quality. Since highly participatory peers are valuable to the network, the peers who service them are also valuable to the network. And since peers that service highly participatory peers are valuable to the network, peers that service them are also valuable to the network, although to a lesser extent.

3 Participation Metrics

We define here some useful notions that are useful in a discussion of participation in P2P networks. Let us define $auth(i, j)$ as the number of authentic files peer i has downloaded from peer j, and $inauth(i, j)$ as the number of inauthentic files peer i has downloaded from peer j. Let us also define a *local participation score*, $s_{ij} = auth(i, j)$ that peer i assigns to peer j that quantifies the quantity and quality of service that peer j has provided to peer i.

The total number of authentic uploads peer j provides to the system is given by $\sum_i auth(i, j)$. This value (the total number of authentic uploads) is the first criteria for participation as outlined in Section 2, and in general is a good participation metric. The sum of a peer's local participation scores $\sum_i [auth(i, j) - inauth(i, j)]$ would also be a good measure of the peer's participation in the network, since it penalizes peers for providing inauthentic files.

However, there are two problems with these metrics. First, neither of these metrics measure the second participation criteria outlined in Section 2. Second, they are difficult to measure, and highly vulnerable to collectives of malicious peers who lie about whether a file is authentic or inauthentic. For this reason, current systems (i.e. [9]) use as a participation metric the total number of uploads peer i provides to the system: $\sum_i [auth(i, j) + inauth(i, j)]$. The clear problem with this metric is that it rewards peers who provide many inauthentic files to the network.

In the next section, we present EigenTrust [7], a scoring system introduced in the context of reputation management that has been shown to be robust to collectives of malicious peers. In this work, we show that EigenTrust scores are also a good measure of participation.

4 EigenTrust

The basic concept behind EigenTrust is that each peer j is assigned a *global participation value*, or *EigenTrust score*, that is given by the sum of the local participation values assigned to peer j by the peers who have interacted with it, weighted by the global participation values of those assigning peers. Therefore, if a collective of malicious peers want to artificially boost the score of peer i, they must have high local participation scores in order to do it. That is, they must upload many authentic files themselves, and thus do service to the network.

We use as the basis for EigenTrust the local participation values s_{ij} defined above.

In order to use local participation values in a working system, it is necessary to normalize them in some manner. Otherwise, malicious peers can assign arbitrarily high

local participation values to other malicious peers, and arbitrarily low local participation values to good peers, easily subverting the system. We define a *normalized local participation value*, c_{ij}, as follows:

$$c_{ij} = \frac{\max(s_{ij}, 0)}{\sum_j \max(s_{ij}, 0)} \tag{1}$$

This ensures that all values will be between 0 and 1. (Notice that if $\sum_j \max(s_{ij}) = 0$, then c_{ij} is undefined. We address this case in [7].) There are some drawbacks to normalizing in this manner. For one, the normalized participation values do not distinguish between a peer with whom peer i did not interact and a peer with whom peer i has had poor experience. Also, these c_{ij} values are relative, and there is no absolute interpretation. That is, if $c_{ij} = c_{ik}$, we know that peer j has the same participation as peer k in the eyes of peer i, but we don't know if both of them are very participatory, or if both of them are mediocre. However, we choose to normalize the local participation values in this manner because it leads to an elegant probabilistic model, and allows us to perform the computation that we describe below without renormalizing the global participation values at each iteration (which is prohibitively costly in a large distributed environment). Furthermore, we are still able to achieve substantially good results despite the drawbacks mentioned above.

Now, we wish to define the global participation value t_j of peer j to be the sum of the local participation values c_{ij} assigned to peer j by the peers i that have interacted with peer j, weighted by the global participation values of those peers i. That is:

$$t_j = \sum_i c_{ij} t_i \tag{2}$$

If we define the matrix $C = [c_{ij}]$ and the vector $\boldsymbol{t}$ to be the vector containing the values t_j, Equation 2 can be written in matrix notation as:

$$\boldsymbol{t} = C^T \boldsymbol{t} \tag{3}$$

In other words, $\boldsymbol{t}$ is the eigenvector of C^T corresponding to eigenvalue 1. Notice that, because of the way we defined the values c_{ij} in Equation 1, the elements of C are all non-negative, and the rows of C sum to 1. Thus, C is a stochastic matrix, and the principal eigenvalue of C^T is 1.

We may use the famous Power Method [5] to find the principal eigenvector $\boldsymbol{t}$ of C, as shown in Algorithm 1 (provided that C is ergodic. In [7], we show that C can always be made ergodic).

This is the basic EigenTrust algorithm. It is important to address how the EigenTrust scores, (or any type of participation scores) are stored and managed in a distributed environment in the absence of a trusted centralized server. One simple way would be for each peer to store and adjust its own score, and report its score when asked by another peer. However, it is easy in this case for malicious peers to misreport their scores. In [7], this problem is addressed by assigning one or more anonymous *score managers* to each peer. These score managers are responsible for storing and managing the scores of their children.

$\boldsymbol{t}^{(0)} = \boldsymbol{e}$;
repeat
  $\boldsymbol{t}^{(k+1)} = C^T \boldsymbol{t}^{(k)}$;
  $\delta = ||t^{(k+1)} - t^k||$;
until $\delta < \epsilon$;

Algorithm 1: Simple non-distributed algorithm

We present the details of the distributed EigenTrust algorithm, and show that it is robust to various threat scenarios, in a companion report [7]. In this work, we are more concerned with how well the EigenTrust scores reflect participation as delineated by the criteria above, and with how to structure score-based incentives in a reasonable manner.

5 EigenTrust and Participation

To test how well a peer's EigenTrust score reflects it's participation level, we simulate a P2P network in the manner described in Appendix 1. The simulator described in Appendix 1 is an event-driven simulator that proceeds by query cycles. At each query cycle, peers submit and respond to queries according to certain distributions over peers' interests and the files peers share, and download files from peers who respond to their queries. Freeriders and malicious peers sharing inauthentic files are modeled as well as active participatory peers.

In Figure 1 we plot each peer in the network on a graph where the x-axis represents the EigenTrust score of the peer, and the y-axis represents the number of authentic uploads that the peer provides in a given timespan (15 query cycles). Notice that the EigenTrust score is correlated with the number of authentic uploads; those peers that provide many authentic uploads also have high EigenTrust score. The correlation coefficient is 0.97 indicating a close relationship between the number of authentic uploads and the EigenTrust score.

In order to give the reader further intuition on EigenTrust scores, we examine the following simple example users in our P2P simulation. These users, and their characteristics, are described below and summarized in Table 1.

Angela is an active participator and shares many popular files across many content categories. Bob is an average user who shares a moderate number of files, many of them popular, some of them unpopular, from a couple of content categories. Corcoran is an occasional user who shares a few files, most of them popular. David is an eccentric user, sharing many obscure files that few people want. Ebeniezer is a freerider, who doesn't share any files. And Forster is a malicious user, who shares many corrupt files.

Based on these descriptions, Angela is the most active participator, followed by Bob, Corcoran, David, Ebeniezer, and Forster, in that order. Table 1 shows that EigenTrust captures this ranking.

Notice that the EigenTrust scores of Ebeniezer the freerider and Forster the malicious peer are both 0. One may wish for a scoring system in which the malicious peer gets a lower score than a freerider. However, it should be noted that, due to the ease of entry in P2P networks, giving a malicious peer a score below that of a freerider is not

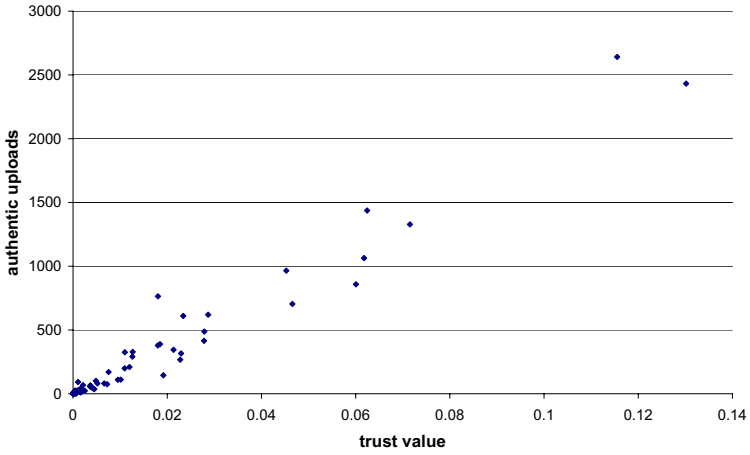

Fig. 1. Correlation between EigenTrust scores (y-axis) and the number of authentic uploads (x-axis). Each point represents a peer.

very effective. A malicious user with a poor score may simply create a new peer and enter the network as a new freerider.

	# Content Cat.	# Files	Popularity	ET Score	ET Rank
ANGELA	10	1,000	Typical	.02875	1
BOB	3	100	Typical	.00462	2
CORCORAN	3	50	Typical	.00188	3
DAVID	3	200	Unpopular	.00115	4
EBENIEZER	0	0	-	0	5 (tie)
FORSTER	10	2,000	Malicious	0	5 (tie)

Table 1. EigenTrust vs. other rankings for each of our seven sample users. Each of these users has the same uptime (20%), and the same number of connections (10).

We also examine the following pairs of peers, where the first member of each pair is more participatory in a different way. Again, we use these examples to give the reader intuition on the behavior of EigenTrust scores.

1. The first scenario involves Tim and Jim. Tim and Jim share the exact same files, except Tim is always online while Jim turns his computer off at night.
2. The second scenario involves Stan and Jan. Stan and Jan share files in the same content categories. However, Stan shares twice as many files as Jan.
3. The third scenario involves Stephanie and Bethany. Stephanie and Bethany share five files each from the same content category. However, Stephanie shares five very popular files, and Bethany shares five unpopular files.
4. The final scenario involves Mario and Luigi. Mario and Luigi share the same number of files, but Mario has a diverse collection, sharing files in many content categories, while Luigi has narrow interests, and shares files from only one content category.

Table 2 compares the EigenTrust scores of each of these pairs of peers, and shows that each of these characteristics we tested (uptime, number of shared files, popularity of shared files, and diversity of shared files) are reflected in the EigenTrust score of a peer. For example, Tim (who shares the same files as Jim, but has a greater uptime) has a greater number of authentic uploads, and a greater EigenTrust score, than Jim.

	# Content Cat.	# Files	Popularity	# Connections	Uptime	# Auth. Uploads	ET Score
TIM	3	1000	Typical	10	92%	965	0.045
JIM	3	1000	Typical	10	35%	415	0.028
STAN	3	500	Typical	10	33%	325	0.011
JAN	3	1000	Typical	10	33%	379	0.018
STEPHANIE	1	5	5 Most Popular	10	35%	15	0.00038
BETHANY	1	5	5 Least Popular	10	35%	0	0
MARIO	5	5000	Typical	10	25%	267	0.023
LUIGI	1	5000	Typical	10	25%	81	0.0052

Table 2. Comparing Pairs of Peers

6 Incentives

Our goal is to provide incentives that reward participatory peers and punish freeriders and malicious peers. However, we do not want to punish freeriders so much that they are not able to download files and share them to become participators if they so choose.

Two ways to reward participatory peers is to award them faster download times, and grant them a wider view of the network. In this section, we propose two score-based incentive schemes: a bandwidth incentive scheme and a TTL incentive scheme. Notice that these score-based schemes that we propose are completely general, and may be used with any scoring scheme that gives scores for which peers should be rewarded. (For example, if another scoring system was introduced to measure participation, these incentives may be used to reward those peers deemed participatory for that reason. For this reason, we use the term "participation score" rather than "EigenTrust score" in Sections 6.1 and 6.2.)

6.1 Bandwidth

The first incentive that we propose is to give active participators preference when there is competition for bandwidth. More specifically, we propose that, if peer i and peer j are simultaneously downloading from another peer k, then the bandwidth of peer k is divided between peer i and peer j according to their participation scores. So if Tim and Jim are simultaneously downloading from some peer k, then Tim will get $\frac{S_T}{S_T+S_J} * 100\%$ of peer k's bandwidth (where S_T represents Tim's participation score, and S_J is Jim's participation score), and Jim will get $\frac{S_J}{S_T+S_J} * 100\%$ of peer k's bandwidth.

This also works when more than 2 peers are simultaneously downloading from the same peer. The protocol for this incentive is given in Algorithm 2.

Peer i with available bandwidth b, assigns bandwidth to each peer j downloading from it as follows:
foreach *peer j downloading from peer i* **do**
$bandwidth(\text{peer } j) = \frac{\text{score(peer } j)}{\sum_j \text{score(peer } j)} b$;
end

Algorithm 2: Bandwidth incentive

Notice that if a peer has a participation score of 0, it will get none of peer k's bandwidth if it is competing against other peers for peer k's bandwidth. However, this doesn't exclude that peer from the network, since it is able to download from peers that are not servicing other peers. We show in Section 7.1 that freeriders are not excluded from the network when this incentive is implemented.

Score Management. In the bandwidth incentive, each peer who is servicing more than one peer at the same time should ask for each downloading peer's participation score. In the insecure case, the servicing peer asks the peers for their participation scores themselves. In the secure case, the servicing peer will ask the score managers of each competing peer for its participation score. If the score managers of a peer i disagree about the participation score of peer i, the servicing peer will take the score reported by the majority of score managers.

6.2 TTL

Currently, Gnutella assigns each peer a time-to-live of 7 for each query. Our second incentive is to assign each peer a TTL based on its participation score, giving active participators a wider view of the network. There are many ways to do this, but one simple way would be to give each peer who has an above-average participation score a high TTL (for example, 10), and to give each peer who has a below-average participation score a low TTL (for example, 5). The protocol for this incentive is given in Algorithm 3.

if $score(\textit{peer } j) > mean\_score$ **then**
$TTL(\text{peer j}) = \textit{high-ttl}$;
else
$TTL(\text{peer j}) = \textit{low-ttl}$;
end

Algorithm 3: TTL incentive

One problem here is that each peer must know the average participation score in the network, and explicitly computing this can be prohibitively costly in terms of message complexity, since it would require each peer to know the participation scores of every other peer in the network. However, if EigenTrust scores are used, this isn't a problem. Since the EigenTrust scores in the network sum to 1, the average EigenTrust score for any network is $1/n$, where n is the number of peers in the network (We assume that a peer either knows or can approximate the number of peers in the network. Methods to

approximate the number of peers in a P2P network are presented in [2]). Therefore, each peer can compare its own EigenTrust score to $1/n$, and if its EigenTrust score is greater, the peer may issue a query with a TTL of 5. Otherwise, the peer may issue a query with a TTL of 3. A peer never needs to explicitly compute the average participation score in the network.

Score Management. For the TTL incentive, each peer assigns itself a query TTL based on it's own participation score. However, in this case, there is nothing preventing every peer to give itself a TTL of 10. In the secure case, a peer relaying a query that has a TTL of 10 will ask the score managers of the peer issuing the query what the participation score of the issuing peer is. If the participation score of the issuing peer is below average, the relaying peer will not forward the query. Notice that this does not address the scenario where a malicious collective collaborates to forward queries regardless of the participation scores of its members. However, the current Gnutella protocol for forwarding queries does not address this either.

6.3 Topology

Another possible incentive is changing the topology of the network based on participation scores. The basic idea behind participation-based topologies is to connect peers with high participation scores to one another to form the hub of the P2P network. Those peers with lower EigenTrust scores should be connected off of this hub. A benefit of this design that is immediately seen is that peers who participate are rewarded by being connected directly to other participatory peers.

Another benefit of this design is that, if we assume that people who share a lot of files also query a lot (or even if we assume that *some* of the people who share a lot of files also query a lot), then connecting peers with high participation scores together makes the network more efficient, since you are connecting people who query a lot with people who share a lot.

There are further benefits if EigenTrust scores are used as the participation score. We can show that a peer's EigenTrust score reflects both the connectivity of a peer and its bandwidth (because if a peer has high bandwidth and is connected to many peers, it is likely to share files to more users, and thus get a high EigenTrust score.) Therefore, connecting peers with high EigenTrust scores together at the core of a P2P topology makes a natural super-peer network.

And finally, since EigenTrust is also also a measure of trustworthiness of peers, EigenTrust-based topologies will move malicious peers to the fringe of the network, limiting the interactions of "good" peers with malicious peers.

We leave a concrete proposal for participation-based topologies to future work, and we do not discuss it further in this paper.

6.4 EigenTrust as an Inherent Incentive

It is worth noting that the EigenTrust algorithm is itself an incentive for participating, since freeriders and malicious peers have little say in determining the participation values of other peers, while highly participatory peers have much say in determining the participation values of other peers. Since rewards may be given to those peers who have high EigenTrust scores, participating in the network gives a peer more say in which peers are rewarded and which peers aren't.

7 Experiments

We have shown in Section 5 that EigenTrust scores are a good measure of participation. Our task in this section is to show that the proposed incentives achieve their stated goals: to reward participatory peers with faster download times and a wider view of the network without completely excluding less active peers from the network. Again, we use our sample peers to give the reader intuition on how these incentives reward peers in the network.

7.1 Bandwidth

The bandwidth incentive aims to reward peers with faster downloads. To test this, we again examine the sample peers Angela et. al., and measure their average download speeds in our simulations with the bandwidth incentive implemented.

In the bandwidth simulation, all peers are assigned an upstream and downstream bandwidth of 1000 kbps. The average file size is 3.7 MB. Peers continuously issue queries as long as they have at least 16 kbps downstream bandwidth available. If a peer requests a download from another peer and is assigned a download rate of less than 16 kbps, it chooses another download source and tries again. The table distinguishes among moderate and premium users. Premium users on average have a higher download rate. The delta is not too significant – this is due to the fact that peers choose different download sources when being assigned a low download rate. Peers with low trust values thus tend to download from peers with low trust values, whereas peers with high trust values (premium users) download more from each other. However, peers with low trust values have to try hard to establish a download connection: The table shows the retry quotient which is computed as number of refused download attempts divided by the total number of download attempts. Premium users are usually admitted to a download from a high-quality peer, normal users sometimes need to try several times (they always attempt to download from high-trust value peers first).

Bandwidth Incentive

	EigenTrust Score	#Peers	Retry quotient	Average download rate
MODERATE USERS	$< \frac{1}{N}$	69	0.7	x1
PREMIUM USERS	$> \frac{1}{N}$	31	0.08	x1.4

Table 3. Retry quotient and average download rate for moderate and premium users in a network using the bandwidth incentive.

Retry quotient and average download rate for normal and premium users

7.2 TTL

Again, the important issue to investigate is whether this incentive compensates participators enough to be a useful incentive while giving nonparticipators enough resources so that they have the option of becoming participators if they so choose.

To do this, we simulate 15 query cycles with the TTL incentive scheme activated as in Algorithm 3. Since this is a small network (100 peers), we define the default TTL to be 4 (peers can reach 75 other peers in this network on average with a TTL of 4). In this case, we define *high-ttl* to be 5, and *high-ttl* to be 3.

For reporting our results, we split the peers into two groups: premium users (those peers with EigenTrust scores larger than the average EigenTrust score $\frac{1}{N}$), and moderate users (those peers with EigenTrust scores larger than the average EigenTrust score $\frac{1}{N}$).

Table 4 shows the number of premium users and moderate users and the average number of peers within their respective TTL range. Note that activating the TTL incentive scheme with these settings decreases the query load on the network while beefing up the service levels for premium users, which is a very desirable result: With the TTL scheme switched on, $77 * 27 + 23 * 98 = 4367$ query messages will be generated throughout the network when all peers issue a query. With the TTL scheme switched off, $100 * 75 = 7500$ messages will be generated in the same process. Also, notice that even the freeriders will not be excluded from the network, as they receive a TTL of 3 for their queries.

		TTL Incentive		**No Incentive**	
	EigenTrust Score	#Peers	#Peers in TTL Range	#Peers	#Peers in TTL Range
MODERATE USERS	$< \frac{1}{N}$	77	27	74	75
PREMIUM USERS	$> \frac{1}{N}$	23	98	26	75

Table 4. The TTL, average number of peers reached (PR), and average number of responses per query (RPQ) for each peer using the TTL incentive in conjunction with the EigenTrust Score (left) and with no incentive (right).

8 Conclusion

Two main results are presented in this paper: First, we show that a peer's EigenTrust score is a good participation metric according to three natural participation criteria. Second, we present two incentive protocols based on these scores, and show that they reward participatory peers without completely excluding non-participatory peers.

References

1. E. Adar and B. Huberman. Free riding on gnutella. *First Monday*, 5(10), October 2000.
2. M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating Aggregates on a Peer-to-Peer Network, 2003.
3. A. Crespo and H. Garcia-Molina. Semantic Overlay Networks. 2002.
4. P. Golle, K. Leyton-Brown, I. Mironov, and M. Lillibridge. Incentives for sharing in peer-to-peer networks. In *WELCOM01*, 2001.
5. G. Golub and C. V. Loan. *Matrix Computations*. Johns Hopkins Press, 1996.
6. B. Horne, B. Pinkas, and T. Sandler. Escrow services and incentives in peer-to-peer networks. In *EC01*, 2001.
7. S. Kamvar, M. Schlosser, and H. Garcia-Molina. EigenTrust: Reputation Management in P2P Networks. In *WWW 2003*, 2003.
8. S. D. Kamvar and M. T. Schlosser. Simulating P2P Networks. Technical report, Stanford University, 2002.
9. KaZaA website. www.kazaa.com.
10. M. Ripeanu and I. Foster. Mapping the Gnutella Network - Macroscopic Properties of Large-scale P2P Networks and Implications for System Design. In *Internet Computing Journal 6(1)*, 2002.
11. S. Saroiu, P. K. Gummadi, and S. D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *MMCN '02*, 2002.

Appendix 1: Simulation

Our findings are based on simulations of a P2P network model which we describe here.

Network model. We consider a typical P2P network: Interconnected, file-sharing peers are able to issue queries for files, peers can respond to queries, and files can be transferred between two peers to conclude a search process. When a query is issued by a peer, it is propagated by broadcast with hop-count horizon throughout the network (in the usual Gnutella way), peers which receive the query forward it and check if they are able to respond to it. We interconnect peers by a power-law network, a type of network prevalent in real-world P2P networks [10].

Node model. Our network consists of participatory peers (peers who share files) and freeriders (peers who don't share files). Those peers who share files share files according to the content distribution model described below. In some experiments, we add a third type of peer: malicious peers, who participate in the network to undermine the network. These peers share inauthentic files, and assign high local participation scores to each other.

Content distribution model. Interactions between peers – i.e., which queries are issued and which queries are answered by given peers – are computed based on a probabilistic content distribution model. The detailed model will not be described here, it is presented in [8]. Briefly, peers are assumed to be interested in a subset of the total available content in the network, i.e., each peer initially picks a number of content categories and shares files only in these categories. Reference [3] has shown that files shared in a P2P network are often clustered by content categories. Also, we assume that within one content category files with different popularities exist, governed by a Zipf distribution. When our simulator generates a query, it does not generate a search string. Instead, it generates the category and rank (or popularity) of the file that will satisfy the query. The category and rank are based on Zipf distributions. Each peer that receives the query checks if it supports the category and if it shares the file. The number of files shared by peers and other distributions used in the model are taken from measurements in real-world P2P networks [11].

Simulation execution. The simulation of a network proceeds in simulation cycles: Each simulation cycle is subdivided into a number of query cycles. In each query cycle, a peer i in the network may be actively issuing a query, inactive, or even down and not responding to queries passing by. Upon issuing a query, a peer waits for incoming responses, selects a download source among those nodes that responded and starts downloading the file. The latter two steps are repeated until a peer has properly received a good copy of the file that it has been seeking. Upon the conclusion of each simulation cycle, the global participation score computation begins. Statistics are collected at each node, in particular, we are interested in the number of authentic and inauthentic up- and downloads of each node. We run an experiment until we see convergence to a steady state (to be defined in the descriptions of the experiments), initial transient states are excluded from the data. Each experiment is run several times and the results of all runs are averaged. The base settings that apply for most of the experiments are summarized in Table 5. The settings represent a fairly small network to make our simulations tractable. However, we have experimented with larger networks in some instances and our conclusions continue to hold.

Network	# of good peers	60
	# of malicious peers	42
	# of initial neighbors of good peers	2
	# of initial neighbors of malicious peers	10
	# of initial neighbors of key peers	10
	# Time-to-live for query messages	4
Content Distribution	# of distinct files at good peer i	file distribution in [11]
	set of content categories supported by good peer i	Zipf distribution over 20 content categories
	# of distinct files at good peer i in category j	uniform random distribution over peer i's total number of distinct files
	top % of queries for most popular categories and files malicious peers respond to	20%
	% of time peer i is up and processing queries	uniform random distribution over [0%, 100%]
	% of up-time good peer i issues queries	uniform random distribution over [0%, 50%]

Table 5. Simulation settings