

BARTERCAST: A PRACTICAL APPROACH TO PREVENT LAZY FREERIDING IN P2P NETWORKS

M. Meulpolder, J.A. Pouwelse, D.H.J. Epema, H.J. Sips

Parallel and Distributed Systems Group
Department of Computer Science, Delft University of Technology, the Netherlands
m.meulpolder@tudelft.nl

ABSTRACT

A well-known problem in P2P systems is *freeriding*, where users do not share content if there is no incentive to do so. In this paper, we distinguish *lazy freeriders* that are merely reluctant to share but follow the protocol, versus *die-hard freeriders* that employ sophisticated methods to subvert the protocol. Existing incentive designs often provide theoretically attractive resistance against die-hard freeriding, yet are rarely deployed in real networks because of practical infeasibility. Meanwhile, real communities benefit greatly from prevention of lazy freeriding, but have only centralized technology available to do so. We present a lightweight, fully distributed mechanism called BARTERCAST that prevents lazy freeriding and is deployed in practice. BarterCast uses a *maxflow* reputation algorithm based on a peer's private history of its data exchanges as well as indirect information received from other peers. We assess different reputation policies under realistic, trace-based community conditions and show that our mechanism is consistent and effective, even when significant fractions of peers spread false information. Furthermore, we present results of the deployment of BarterCast in the BitTorrent-based Tribler network which currently has thousands of users worldwide.

1. INTRODUCTION

The performance of P2P file sharing systems depends on the contribution of resources by their participating users. Whereas some users share their content altruistically, other users only share when there is a direct or indirect incentive to do so and might *freeride* if such an incentive is not present. Freeriding is a well-known issue in P2P research; its effects have been empirically measured [1, 20] and extensively analyzed [2, 9, 12, 15]. Many incentive mechanisms and reputation systems have been proposed to induce cooperation and reduce freeriding [8, 10, 11, 13, 21]. Much attention in these proposals is given to security issues such as cheating, whitewashing, collusion, and sybil attacks [6]. Meanwhile, deployed systems use straightforward central server technology to punish and reward users. Maze

[16] and eMule [7] use point/credit systems so that users with a high balance can be given precedence by others. Private BitTorrent communities (e.g., Demonoid, TVTorrents, Filelist) are more strict and provide long term incentives using centralized *sharing-ratio enforcement*, i.e., banning of peers that upload too little compared to what they download. Andrade et al. [2] show that in private BitTorrent communities the download performance is much higher than in public communities due much larger numbers of peers that are sharing their content. In addition, the content collections in such communities are more balanced and offer a higher performance not only for popular content but also for content that is rare and not well-known. However, such centralized systems have a central point of failure, a considerable administration and management overhead, and require users' trust in an unknown authority with respect to privacy sensitive information. Our goal is to provide *fully distributed freeriding prevention* that is effective but also practically feasible, providing P2P systems with the superior performance of private BitTorrent communities without the need of any central technology.

Virtually all distributed solutions presented by the research community until now have remained paper designs and are not deployed in any real file-sharing community. An important reason for this is that most research has focused on finding 'waterproof' distributed solutions that are resistant against all forms of cheating. Such solutions sacrifice practical feasibility for attractive theoretical properties. We let go of this present obsession with cheating, and instead focus on a more pragmatic approach. We make a distinction between *lazy freeriders* that do not share a file after they have finished downloading versus *die-hard freeriders* that create or obtain cheating clients to prevent uploading at all. This distinction is supported by the following observations: (i) communities in the real world do work well with millions of users while they are not resistant against all forms of cheating at all [2]; (ii) only a small fraction of the users actually uses sophisticated measures to subvert the protocol to freeride [22]; (iii) despite the existence of cheating clients [14, 15, 18], most participants of file-sharing communities use clients that follow the protocol [3]. Research until now

has given most of its attention to die-hard freeriding, which has hampered the development of practical alternatives for currently deployed centralized technology. We follow another approach by focusing on lazy freeriding and present feasible technology that actually works in real systems.

In this paper, we provide the following contributions: (1) We present BARTERCAST, a *fully distributed* mechanism for lazy freeriding prevention, based on the *maxflow algorithm* [5], that is lightweight, practically feasible, and deployed in a real P2P file-sharing network; (2) We assess two policies for the integration of BarterCast with BitTorrent and show how these policies result in a strong incentive for freeriders to become sharers, even when up to 18% of the peers spread false information; (3) We present extensive simulation results that demonstrate the consistency and effectiveness of our maxflow-based approach using traces of a popular, real-world BitTorrent community. We hereby significantly extend earlier proposals of maxflow-based reputation and take it from a merely theoretical idea to a fully practical and realistic approach; (4) We present results of the deployment of BarterCast in Tribler [19] (www.tribler.org), a BitTorrent-based P2P client that is used by thousands of active users worldwide.

2. RELATED WORK

A wide range of research addresses the freeriding problem and its properties, and many different incentive mechanisms and reputation systems have been proposed. Gupta et al. [10] present a reputation system that is partially distributed, but relies on the authority of a single agent that stores peer reputations. As a mechanism for multiple agents is not discussed, the presented solution in essence remains centralized. Kamvar et al. [11] present EigenTrust, an algorithm for reputation management in P2P networks that is based on distributed computing of a globally consistent trust vector for every peer in the system. EigenTrust's theoretical properties are attractive. On the downside, EigenTrust relies on pre-trusted peers for convergence and its aim for global consistency assumes a rigid network of peers. Karma [21] is a system in which sets of bank nodes keep track of the transaction balance of peers. The approach relies on a DHT-based structure of allocating bank nodes and cryptographic mechanisms for security. BitTorrent [4] uses a tit-for-tat policy with short-term, direct incentive properties. While tit-for-tat is successful in short term transactions, the mechanism offers no incentives for long-term sharing of content. Lian et al. [13] identify key disadvantages of both EigenTrust and tit-for-tat and propose a hybrid model in between the two. Recently, Piatek et al. [17] have presented a reputation mechanism based on intermediaries that attest to the behaviour of others. The suggested mechanism relies heavily on the availability and connectability of the intermediaries at the time of a transaction; it requires their response before

a transaction can take place. This is not robust in the dynamic file-sharing networks of today. Feldman et al. [8, 9] analyze freeriding and whitewashing and propose some incentive techniques to deal with them. While they briefly introduce a reputation mechanism based on maxflow, they implicitly assume that all nodes have a complete view of the network, which is highly unrealistic in large, dynamic communities. Moreover, the authors omit any discussion on how nodes acquire this information. They provide a simple reputation metric which does not distinguish 'bad' peers from newcomers, and do not include any experimental results on the consistency and effectiveness of maxflow in practice. While their proposal offers a direction, their contribution stays completely theoretical. In general, we are not aware of any practically feasible distributed reputation mechanism that is tested under realistic, trace-based community conditions, and deployed in a real-world P2P network.

3. THE BARTERCAST PROTOCOL

In this section we present BARTERCAST, a lightweight message exchange protocol integrated with a maxflow-based reputation mechanism. We present a reputation metric that allows a distinction between sharers, freeriders, and neutral peers. The message exchange protocol provides peers with a (partial) view of the network in terms of the data transfers that have occurred between peers. We will describe the general idea of our approach, the maxflow mechanism, and the message exchange protocol.

3.1. General idea

In social networks in the real world, a person has a sense of reputation regarding another person which is usually based on two factors: (1) direct experience, and (2) information about this person obtained from other people. However, the evaluation of information obtained from a particular source is constrained by the reputation this source has at the evaluating party. As a result, each person has a personal *web of reputation* (or *web of trust*) in which reputation is *subjective* and based on incomplete information. To a certain extent, this approach is vulnerable to abuse (e.g., by lying), and can lead to incorrect decisions. Yet it is deeply ingrained in our social interactions, and seems to work fairly well. It is obvious that if all people had to reach global consensus about the reputation of every single person in the world, life would become very complicated.

We apply this same concept of reputation to file-sharing networks. In this analogy, we regard as 'direct experience' the aggregated amount of *service* a peer has received from another peer in the past. The network can then be regarded as a graph with the peers as nodes, and directed edges that represent the aggregated amount of service. Peers exchange information with others about their direct experience so that

each peer can build its own, local representation of this graph. In order to identify freeriders, peers have to evaluate the *upload* and *download* behaviour of other peers. Our measure of performed service is therefore the *total number of bytes* transferred from one peer to another peer in the past.

3.2. The maxflow algorithm

The maxflow algorithm is a classic algorithm that has applications in a variety of fields such as logistics, financial analysis, and computer networking. In [5], the original Ford-Fulkerson maxflow algorithm is described, along with examples, complexity analysis, and a proof that the resulting flow is indeed maximal. Given is a graph G with capacity $c(i, j)$ and flow $f(i, j) = 0$ for every edge (i, j) from i to j . The maxflow algorithm (Algorithm 1) returns the maximal flow from a given source node s to a target node t . For finding the paths in line 5 we use a common depth-first search. After each iteration in the while loop of the algorithm, the *residual network* $G_f(V, E)$ is computed as the network with capacities $c_f(i, j) = c(i, j) - f(i, j)$ and no flow on the edges. This residual network represents the *remaining capacity* in the network after certain flows have been applied. Note that in order to find the maximum flow over all paths, it might be necessary during some stages of the algorithm to decrease an earlier applied flow over a certain path, which effectively corresponds to increasing a flow in the ‘reverse direction’. The code at line 9 ensures the consistency of the increased/decreased flows with the capacities in the residual network (for more details see [5]). When consistent flows over all possible paths have been found, the maximum flow from s to t is equal to the total incoming flow at the target node t , computed at lines 14-16.

Algorithm 1 The Ford-Fulkerson maxflow algorithm [5].

```

1: for all  $(i, j) \in E$  do
2:    $f(i, j) \leftarrow 0$ 
3: end for
4: initialize  $G_f(V, E)$ 
5: while  $\exists$  path  $p$  from  $s$  to  $t$  in  $G_f$ , such that  $c_f(i, j) > 0$ 
    $\forall (i, j) \in p$  do
6:    $c_f(p) = \min\{c_f(i, j) | (i, j) \in p\}$ 
7:   for each  $(i, j) \in p$  do
8:      $f(i, j) \leftarrow f(i, j) + c_f(p)$ 
9:      $f(j, i) \leftarrow f(j, i) - c_f(p)$ 
10:  end for
11:  update  $G_f(V, E)$ 
12: end while
13:  $maxflow = 0$ 
14: for all  $(i \neq t)$  do
15:    $maxflow \leftarrow maxflow + f(i, t)$ 
16: end for
17: return  $maxflow$ 

```

We apply the maxflow algorithm in a completely different context than what it was originally meant for. In our case, the ‘capacity’ $c(i, j)$ is defined as the total number of bytes that peer i has uploaded to peer j in the past. The concept of a ‘flow’ no longer denotes a dynamic, fluid-like process, but is used as a measure of ‘indirect service’ in a network of direct transactions. Despite this counter-intuitive application of the maxflow algorithm, it has a very interesting property. When the maxflow based on aggregated bytes is used by a peer to evaluate another peer, the evaluation of peers of which no direct service is received is constrained by the amount of service that is received *directly*. This has the effect that the influence of incorrect information is to a certain extent limited, yet without the need for global consensus. Furthermore, the more transactions a peer has performed, the more well-informed its perceived reputation of others will become. These effects mimics the subjective notion of reputation in real social networks and is what we aim for in our approach.

To limit the practical computation overhead, our implementation only regards paths with a maximum length of two. This simplification is very reasonable in P2P file-sharing networks, since measurements such as in [17] show that P2P file-sharing networks demonstrate a *small-world effect* where 98% of peer pairs either exchanged data directly or exchanged data with a common third party.

3.3. Reputation metric

We introduce a reputation metric based on maxflow, with which a peer can evaluate and compare the contribution of other peers. We define the *subjective reputation* $R_i(j)$ of peer j at peer i as follows:

$$R_i(j) = \frac{\arctan(maxflow(j, i) - maxflow(i, j))}{\pi/2} \quad (1)$$

The resulting reputation has a value between -1 and 1. The reputation value can be used to make a distinction between a negative, a positive, and a neutral contribution of a peer. The scaling provided by the *arctan* function has the effect that the difference between, for instance, 0 and 100 MB is more significant than the difference between 1000 MB and 1100 MB. This ensures that a modest contribution of a new (or neutral) peer significantly effects its reputation, and is not dwarfed in comparison with the most active peers.

3.4. Message exchange

Peers need to build a local representation of the network as input for the maxflow algorithm. They obtain this information in two ways. First of all, each peer keeps a *private history* of the transfers it had with other peers. The private history at peer i is a table where an entry $(j, up, down)$ is a record of the number of bytes peer i has uploaded to,

respectively downloaded from, peer j . Secondly, a peer obtains information about the rest of the network by exchanging a selection of its private history with others using messages. Peer i selects for its messages the records of the N_h peers with the highest upload to i as well as the N_r peers most recently seen by i . In this way, each peer builds up a *subjective shared history* about the transfers in the network. The private history and subjective shared history then represent a subjective, local graph which is used as input for the maxflow algorithm. In a large system the local view each peer has of the network will obviously not be complete, and not all information will be up-to-date. However, we will show in our experiments that this subjective approach based on partial information is yet very effective in distinguishing lazy freeriders from non-freeriders. Of course, a die-hard freerider j with protocol knowledge and skills can choose to spread false information in an attempt to boost its reputation at a peer i . However, maxflow has the natural property that the value of $maxflow(j, i)$ is always constrained by i 's incoming edges. The information about *these* edges is derived from peer i 's private history which itself cannot be manipulated by others. In our current system we do not aim to be fully resistant to die-hard cheating techniques. However, we will show that our system is robust against significant fractions of peers that are not sending any message at all, as well as against significant fractions of peers that are lying. We assume that peers can discover other peers by using a *Peer Sampling Service (PSS)*. The actual implementation of such a service is transparent to BarterCast, and it is sufficient here to note that distributed PSS protocols are feasible and do exist. We use the decentralized BuddyCast epidemic protocol that is integrated in Tribler [19].

3.5. Whitewashing

A common concern in reputation systems is *whitewashing*, i.e., users can get rid of a negative reputation easily by assuming a new (cheap) identity. As observed in [9], there are only two ways to counter whitewashing. The first is to create unforgeable, strong identities that are one-on-one linked to a user, for example by uniquely tying an identity to an IP-address. If such identities are not present, newcomers are undistinguishable from whitewashers and the only approach [9] is to impose a penalty on all newcomers. This penalty can be static or it can be determined dynamically using an *adaptive stranger policy*. What constitutes an effective whitewashing penalty is highly dependent on the specific freerider policy and the details of the P2P protocol used. In our current implementation we assume for now that the P2P client software can create a machine-dependent permanent identifier which takes considerable programming skills to change. The Tribler P2P client employs such permanent identifiers in practice. However, in future work we will assess policies that do not depend on strong identities.

4. REPUTATION-BASED BITTORRENT

In this section we discuss our application of the BarterCast reputation system in BitTorrent. BitTorrent currently only implements a tit-for-tat policy which gives downloaders in a single swarm an incentive to upload to each other. However, there is no incentive to continue sharing the file after the download has finished. Ironically, it is even disadvantageous to share, since the consumed upload bandwidth cannot be used to do tit-for-tat in other downloads, which makes these downloads slower. We will first give a short overview of the BitTorrent tit-for-tat policy, and after that describe two policies in which long term reputation is taken into account.

4.1. BitTorrent tit-for-tat

The normal BitTorrent protocol maintains a limited number of simultaneous upload slots (usually 4-7 depending on the implementation). Peers that do not yet have the complete file (*leechers*), assign their slots to those peers that currently provide the highest upload rate in return, determined periodically. Peers that have the complete file (*seeders*) assign their upload slots to those peers that have the highest download rate. Peers that get a slot are called *unchoked*, while the other peers are *choked*. Furthermore, there is one extra slot for *optimistic unchoking*. This slot is assigned via a 30 seconds round-robin shift over all the interested peers regardless of their upload rate. The protocol therefore creates a tit-for-tat data exchange based on the short-term behaviour of a peer (i.e., the bandwidth it provides in return). Due to optimistic unchoking, new peers have a chance to obtain their first pieces of data and bootstrap the process.

4.2. Reputation policies in BitTorrent

When a reputation system such as BarterCast is available, a variety of policies can be designed that use the reputation information in a P2P protocol such as BitTorrent. In our experiments we have assessed two policies that reduce the system-wide performance of lazy freeriders while being easy to include in any BitTorrent implementation. The policies, which use the reputation of Equation (2), are:

rank policy Peers assign optimistic unchoke slots to the interested peers in order of their reputation. Therefore, a peer can not get an upload slot while peers with a higher reputation are also interested and not yet served.

ban policy Peers do not assign any upload slots to peers that have a reputation which is below a certain negative threshold δ .

Of course, many policies can be thought of that make more sophisticated use of the long term reputation provided by BarterCast. Furthermore, various policies can be designed for P2P protocols other than BitTorrent.

5. RESULTS

In this section we present the results of trace-based simulations in which we analyze the effectiveness and accuracy of our reputation system under realistic community conditions. Furthermore, we present measurements of the deployment of BarterCast in the open-source P2P client Tribler [19].

5.1. Simulation setup

We have built a trace-based simulator which incorporates all relevant aspects of BarterCast and BitTorrent. We simulate an epidemic Peer Sampling Service combined with the BarterCast protocol and the BitTorrent protocol. Our BitTorrent simulator follows the protocol at the piece-level, including unchoking, optimistic unchoking, and rarest-first piece picking. We have combined all processes in a trace-based simulation environment. In our simulations, we use network traces from the popular private BitTorrent tracker `filelist.org`. The traces contain detailed behaviour of all peers that were active in the file-sharing network, including uptimes, downtimes, connectability, and file-requests. We have chosen this tracker because it is not possible to scrape such detailed information from most other trackers.

For our simulations we use a subset of $N = 100$ peers active in 10 different swarms during one week. We simulate that 50% of the peers are (lazy) *freeriders* that immediately leave the swarm after finishing a download, while the other 50% are *sharers* that share every downloaded file for 10 hours. The filesizes in the trace range from several tens of megabytes to about one to two gigabytes, representing mostly audio and movie files. Since we do not have information about the real up- and download bandwidths of the peers, we currently simulate common ADSL users with a 3 MBps downlink and a 512 KBps uplink. As these users have very limited uploading capacity, they are likely to economize on sharing, and are therefore the most important target of sharing-ratio enforcement in current file-sharing systems. Finally, in the BarterCast messages we have used $N_h = N_r = 10$.

5.2. Contribution versus reputation

We first compare the real behaviour of a peer with its average reputation in the system. In order to measure a peer's real behaviour, we compute a peer's *net contribution* which is defined as its real total upload minus its real total download during the simulation period. Since the reputation of a peer is subjective with respect to another peer, we compute the *system reputation* \mathbf{R}_i of peer i as the average of the reputations it has at each of the $N - 1$ other peers:

$$\mathbf{R}_i = \frac{1}{N-1} \sum_{j \neq i} R_j(i) \quad (2)$$

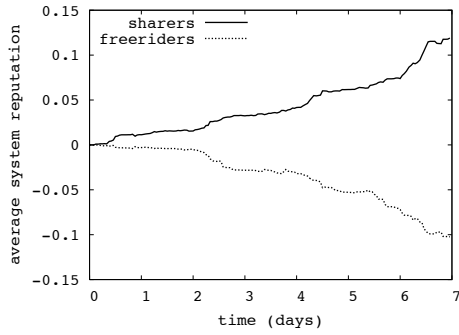
Figure 1 (a) shows a plot of the average system reputation of both the freeriders and the sharers throughout the simulation period of one week. It is visible that the reputations quickly diverge and that freeriders are clearly distinguished from sharers. In Figure 1 (b) the correlation between the net contribution and the system reputation of individual peers is plotted. The relationship between a peer's net contribution and its system reputation is clearly consistent, and indicates that despite the fact that peers have incomplete information about other peers, a peer's resulting reputation in the system is a good representation of its real behaviour.

5.3. Effectiveness of different policies

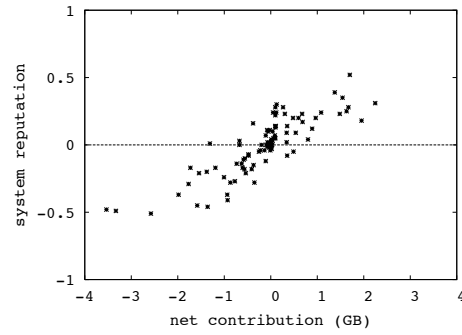
We have assessed both the rank policy where peers with the highest reputation receive the optimistic unchoke slots first, and the ban policy where peers with a reputation below δ are banned. For the ban policy we have run simulations with various values for δ . Figure 2 (a) and (b) compare the rank policy and the ban policy (with $\delta = -0.5$) by monitoring the average download speed of the sharers and the freeriders throughout the simulation period. First of all, we observe that during the first days the average speed of freeriders is higher than that of sharers. This is because freeriders do not allocate any upload bandwidth to the sharing of complete files and therefore have more upload capacity available to do tit-for-tat in their downloads. After some time however, the speed of the sharers increases and the speed of the freeriders decreases under both policies, leading to a significantly higher speed for the sharers. At the end of the simulation period, the freeriders have about 75% of the speed of the sharers if the rank policy is used, while they reach only about 50% of the speed of the sharers if the ban policy is used. The ban policy is therefore clearly superior, i.e., it provides a much stronger disincentive to be a freerider, thereby promoting systems with more sharers, which in its turn implies higher speeds. In Figure 2 (c) the results of the ban policy are compared for various values of δ . The difference between using $\delta = -0.3$ and $\delta = -0.5$ is clearly less significant than the difference between using $\delta = -0.5$ and $\delta = -0.7$.

5.4. Disobeying the protocol

Even though we do not have it as our primary focus to prevent die-hard freeriding behaviour, we have measured what the effect is on a system when peers manipulate the BarterCast message protocol. We have tested two different ways of manipulation: (1) peers *ignore* the message protocol, i.e., they do not send any information at all; (2) peers *lie* in a selfish way by claiming they sent huge amounts of data to other peers and received nothing. For both cases we have measured what the relative speed of all freeriders in the system is as compared to the speed of the sharers. We have run tests with different fractions of disobeying peers, and

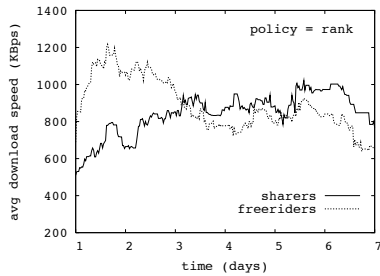


(a)

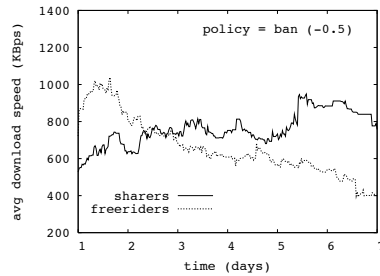


(b)

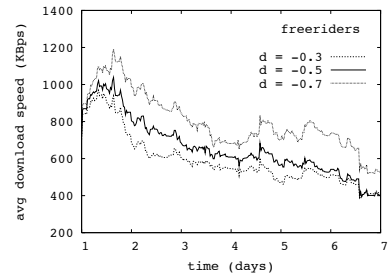
Figure 1: (a) The system reputation in the system for both sharers and freeriders; (b) A scatter plot of the system reputation versus the net contribution of peers (each dot represents one peer).



(a)

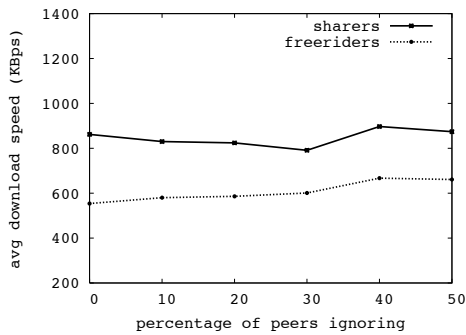


(b)

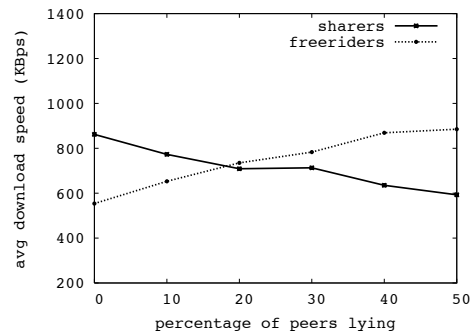


(c)

Figure 2: (a) The average download speed using the *rank policy*; (b) The average download speed using the *ban policy* with δ equal to -0.5; (c) The average download speed of freeriders using the ban policy with different values of δ .



(a)



(b)

Figure 3: The average download speed of both freeriders and sharers in simulations with peers that (a) ignore the message protocol; (b) lie about their contribution.

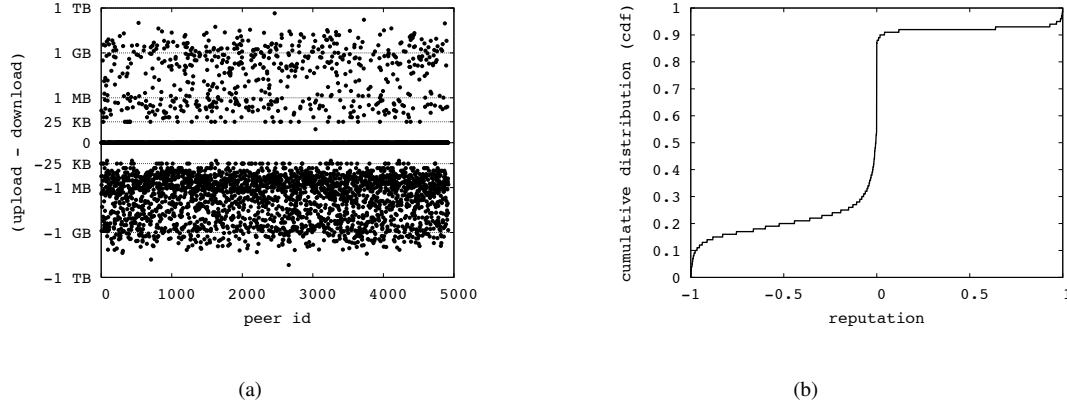


Figure 4: (a) Upload minus download of 5000 peers seen during one month of deployment; (b) Cumulative distribution function of the reputation of these peers, as computed by the peer that performed the measurements.

have assumed that disobeying peers are a random selection from a total of 50% freeriders in the system. We assume that the sharers, as they are cooperative, do obey the protocol. Hence, in our tests, at most 50% of the peers disobey. We use the ban policy with $\delta = -0.5$. In Figure 3 (a), the results are plotted for peers that ignore the protocol. It is clear that this behaviour does not significantly change the effectiveness of our reputation system. The sharers (from which most data comes) base their banning decisions on the information coming from other sharers and from obeying (lazy) freeriders. Apparently this information is sufficient to maintain the lower speed freeriders are intended to have. In Figure 3 (b), the results are plotted for peers that are lying. It is visible (and expected) that the effectiveness of the protocol decreases when the fraction of lying peers goes up. However, for significant fractions of lying peers ($< 18\%$) the protocol is still effective. In real systems, where only a small minority of the users is likely to be able to change their client software, the protocol is therefore definitely promising.

5.5. Deployment results

We have deployed a fully operational version of BarterCast in version 4.2 of Tribler [19], a BitTorrent compatible file-sharing client which has an active network of thousands of users. As we first want to gain understanding of user behavior in this network, the current version not yet enforces a policy to penalize low reputation. We logged all BarterCast messages received by a customized peer participating in the network during the first month after its initial deployment. Figure 4 (a) shows the upload minus download of approximately 5000 peers seen during the measurement by our customized peer. It is clearly visible that a majority of the peers has downloaded more than what they have uploaded. Peers that have a value of exactly zero have most

likely just installed the client without using it for downloading/uploading (yet). One important thing to note is that Tribler clients can participate in any BitTorrent network and exchange data with any other BitTorrent client. Therefore, the total upload of the Tribler peers in our measurement is not necessarily equal to the total download of all of these peers. Figure 4 (b) shows the reputation of the same set of peers as computed by our customized peer. About 40% of the peers have downloaded more than what they have uploaded, while only 10% have uploaded more than what they have downloaded. There are even a few very generous ‘altruists’ visible with tens of gigabytes contribution to the network. The contribution imbalance visible in both graphs clearly demonstrates the need for uploading incentives in the Tribler network. As our measurements are ongoing, we plan to present an extensive range of deployment results in future work in which we will also assess the the fractions of freeriders that turn into sharers when penalizing policies are in place.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have argued for a distinction between *lazy freeriders* and *die-hard freeriders* in P2P systems, based on the observation that many real file-sharing communities and protocols are not completely resistant against cheating but are nevertheless very successful. We have presented a practical, light-weight maxflow-based reputation mechanism called BARTERCAST which allows P2P protocols to penalize peers that freeride and therefore give them an incentive to share. We have also presented concrete policies to adapt this reputation system in BitTorrent and we have validated our claims by presenting the results of simulations based on extensive traces of a real major file-sharing community. It turns out that subjective banning of peers based

on their reputation leads to a significant reduction in speed for freeriders, and therefore an incentive to share. Furthermore, we have assessed the properties of the system under different fractions of peers that do not obey the protocol, and have shown that our approach is still effective with up to 18% of the peers selfishly lying about their contribution. Finally, we have presented results of the first month of deployment of BarterCast in the Tribler P2P client. We plan to present more extensive deployment results in future work based on our current deployed version as well as future versions that enforce reputation policies. Furthermore, we plan to perform simulations with up to 100,000 peers and assess the scalability of our mechanism. We will also explore more sophisticated reputation policies and techniques to prevent die-hard cheating and malicious behaviour, with the constant aim of keeping our mechanism practically feasible.

References

- [1] E. Adar and B. A. Huberman. Free Riding on Gnutella. Technical report, Xerox PARC, August 2000.
- [2] Nazareno Andrade, Miranda Mowbray, Aliandro Lima, Gustavo Wagner, and Matei Ripeanu. Influences on cooperation in bittorrent communities. In *P2PECON '05: Proceedings of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 111–115, New York, NY, USA, 2005. ACM.
- [3] P2p marketshare. <http://torrentfreak.com/p2p-statistics-080426>.
- [4] B. Cohen. Incentives Build Robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer Systems*, Berkeley, USA, May 2003. <http://bittorrent.com>.
- [5] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction to Algorithms*, pages 651–664. MIT Press and McGraw-Hill, second edition, 2001.
- [6] John R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260. Springer-Verlag, 2002.
- [7] Emule. <http://www.emule-project.net>.
- [8] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *Proc. of the 5th ACM conference on Electronic Commerce (EC'04)*, pages 102–111, May 2004.
- [9] Michal Feldman, Christos Papadimitriou, and John Chuang. Free-riding and whitewashing in peer-to-peer systems. In *Proc. PINS*, pages 228–236. ACM Press, 2004.
- [10] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *Proc. of NOSS-DAV'03*, June 2003.
- [11] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *12th WWW conference*, New York, 2003.
- [12] R. Krishnan, M. Smith, Z. Tang, and R. Telang. The virtual commons: why free-riding can be tolerated in peer-to-peer networks. In *Workshop on Information Systems and Economics*, December 2003.
- [13] Qiao Lian, Yu Peng, Mao Yang, Zheng Zhang, Yafei Dai, and Xiaoming Li. Robust incentives via multi-level tit-for-tat. In *Proc. 5th International Workshop on Peer-to-Peer systems (IPTPS)*, 2006.
- [14] N. Liogkas, R. Nelson, E. Kohler, and L. Zhang. Exploiting bittorrent for fun (but not profit). In *Proc. 5th International Workshop on Peer-to-Peer systems (IPTPS)*, 2006.
- [15] Thomas Locher, Patrick Moor, Stefan Schmid, and Roger Wattenhofer. Free riding in bittorrent is cheap. In *Proc. of HotNets-V*, 2006.
- [16] Maze. https://sarwiki.informatik.hu-berlin.de/The_Maze_Peer-To-Peer_System.
- [17] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson. One hop reputations for peer to peer file sharing workloads. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation (NSDI '08)*, San Francisco, CA, USA, pages 1–14, April 2008.
- [18] Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. Do incentives build robustness in bittorrent? In *NSDI'07*, Cambridge, MA, April 2007.
- [19] J.A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D.H.J. Epema, M. Reinders, M.R. van Steen, and H.J. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 19, 2008.
- [20] Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. of Multimedia Computing and Networking 2002 (MMCN '02)*, 2002.
- [21] V. Vishnumurthy, S. Chandrakumar, and E.G. Sirer. Karma: A secure economic framework for peer-to-peer resource sharing. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.
- [22] Manaf Zghaibeh and Kostas G. Anagnostakis. On the impact of p2p incentive mechanisms on user behavior. In *Proc. of the Joint Workshop on The Economics of Networked Systems and Incentive-Based Computing*, 2007.