SIMULATING THE HUMAN FACTOR IN REPUTATION MANAGEMENT SYSTEMS FOR P2P NETWORKS

An Agent Based Model

Guido Boella, Marco Remondino
University of Turin, Department of Computer Science, Cso Svizzera 185, Turin, Italy

Gianluca Tornese
University of Turin, Italy

Keywords: Reputation system, Agent based simulation, Peer to peer network.

Abstract: A compelling problem in peer-to-peer (P2P) networks for file sharing, is the spreading of inauthentic files. To counter this, reputation management systems (RMS) have been introduced. These systems dynamically assign to the users a reputation value, considered in the decision to download files from them. RMS are proven, via simulation, to make P2P networks safe from attacks by malicious peers spreading inauthentic files. But, in large networks of millions of users, non-malicious users get a benefit from sharing inauthentic files due to the credit system. In this paper we show, using agent based simulation, that reputation systems are effective only if there is a widespread cooperation by users in verifying authenticity of files during the download phase, while the size punishment derived by the reputation systems is less relevant. This was not evident in previous works since they make several ideal assumptions about the behaviour of peers who have to verify files to discover inauthentic ones. Agent based simulation allows to study the human factor behind the behaviour of peers, in particular the advantage of spreading inauthentic files, of not checking as soon as possible their authenticity during the download, thus unwillingly cooperating to the spreading of files-point.

1 INTRODUCTION

One of the most compelling problems in peer-to-peer (P2P) networks for file sharing, at least from the point of view of users, is the spreading of inauthentic files; since multimedia files are usually quite large, downloading the wrong ones causes waste of time, of storage space and of bandwidth. The percentage of inauthentic files circulating over a P2P network is high, especially for those resources which are recent and most requested.

Some of the reasons of this problem are the possibility to attack a P2P networks introducing malicious peers and how users exploits the protocol with which the current P2P systems reward the users for uploading files, no matter if they are authentic or not. Since uploading bandwidth is a limited resource and the download priority queues are based on a uploading-credit system to reward the most collaborative peers on the network, some malicious users, if they see that a resource is rare or most wanted, could decide to create an inauthentic file for it, just to have something to share, thus obtaining credits. In this way inauthentic files spread very quickly on the network and those malicious users are never penalized for their behaviour.

To balance the incentive for sharing inauthentic files, reputation management systems (RMS) have been introduced. These systems gather information from users about the authenticity of files downloaded by peers and based on this dynamically assign a reputation to them, used as a value in the decision to download files from them or not. Reputation management systems are proven, via simulation, to make P2P networks safe from attacks by malicious peer, even when forming coalitions.

If attack to P2P network via inauthentic files is a relevant problem, it is not the only one. In particular, in large networks of millions of users attacks are more difficult but users still have a benefit from

Boella G., Remondino M. and Tornese G.
SIMULATING THE HUMAN FACTOR IN REPUTATION MANAGEMENT SYSTEMS FOR P2P NETWORKS - An Agent Based Model.
DOI: 10.5220/0001821200050012
In *Proceedings of the Fifth International Conference on Web Information Systems and Technologies (WEBIST 2009)*, page
ISBN: 978-989-8111-81-4

sharing inauthentic files. RMS have been proved useful against attacks but it is not clear if they can be effective against widespread selfish misbehaviour. The reason is that they make many ideal assumptions about the behaviour of peers who have to verify files to discover inauthentic files: this operation is assumed to be automatic and to have no cost. Moreover, since in current systems files are shared before downloading is completed, peers downloading inauthentic files involuntarily spread inauthentic files if they are not cooperative enough to verify their download as soon as possible. In this paper we address the following research questions:

- Which are the factors controlled by users which determine the correct functioning of reputation management systems in P2P network?
- How to evaluate the role of these factors by means of agent based simulation? Which factors have most impact?

In the present work the creation and spreading of inauthentic files is not considered as an attack, but a way in which some agents try to raise their credits, while not possessing the real resource that's being searched by others. A basic and idealized RMS is introduced, acting as a positive or negative reward for the users. We are not interested in simulating factors like distribution of workload among the peers, how reputation is calculate, bandwidth issues and multiple sources downloading, since our focus is different. Instead, the human factor behind a RMS is considered, e.g. the decision of creating inauthentic files and costs and benefits of verifying files.

To test the limits and effectiveness of a RMS under different user behaviours we use agent based simulation. A multi agent simulation of a P2P network is used, employing reactive agents to model the users. The P2P network is modelled using a graph with a random distribution for the nodes and the links. The RMS is decentralized and each agent has a unique value, valid throughout the whole system. Each agent has a dynamic set of goals; the protocol used for searching is a request flooding one, with a TTL (time to live) set to three.

Under certain conditions, even a simple RMS can dramatically reduce the number of inauthentic files over a P2P system and harshly penalize malicious users, without directly banishing them from the network (ostracism).

The main goal is to discover the malicious users and penalize them for their behaviour, while rewarding the loyal users. By using given values for positive and negative payoffs, malicious users get quickly discovered, in the sense that their reputation, after some simulated turns, gets significantly lower than that of loyal agents. Besides, the unaware agents that unwillingly contributed to spread the inauthentic files (since they didn't check and delete them when they could do that) do not get penalized too much by this system, keeping a reputation value much higher than malicious agents. After few turns the number of inauthentic files over the P2P network is dramatically reduced, and so is the index of their number/total number of requests. This guarantees that this emerging result is not caused by the reduction of the number of requests.

This paper is structured as follows. First in Section “reputation” we discuss reputation management systems, then in Section “model” we present the P2P scenario we use. Section “results” analyzes the parameters and results of simulations. “Conclusions” discusses the results and future work.

2 REPUTATION

It is well known that selfish peers wish to use file sharing services while contributing minimal or no resources themselves: to minimize their cost in bandwidth and CPU utilization freeholders refuse to share files in the network. To incentive sharing of files, credit systems are set up to give more bandwidth or priorities in waiting queues to users who share more files.

The credit mechanism, alas, can be exploited by users who try to get more credits by distributing highly requested files even if they do not have them, it is sufficient to replace them with inauthentic copies. This behaviour can take the form of an explicit creation of inauthentic files or the voluntarily or not voluntarily sharing of downloaded (or still in download) inauthentic files. When a user is downloading a file after a given amount of time he can check its authenticity. If she does not verify, or she does not remove that file, from the perspective of the system she becomes a malicious user. RMS, like EigenTrust (Kamvar et al., 2003), try to lower the problem of inauthentic files by allowing users to give feedback about the cooperativeness of other users and the system uses this feedback to compute a reputation of the different peers in the systems.

In general, a RMS assists agents in choosing a reliable peer (if possible) to transact with when one or more have offered the agent a service or resource. To accomplish this, a RMS collects information on the transactional behaviour of each peer (information gathering), scores and ranks the peers based on expected reliability (scoring/ranking), and

allows the system to take actions against malicious peers while rewarding contributors.

The driving force behind RMS design is to provide a service that mitigates misbehaviour while imposing a minimal costs on the well-behaved users. To that end, it is important to understand the requirements imposed on system design by each of the following: among them the behaviour and expectations of typical good users, and the goals and attacks of adversaries.

Usability of a systems consists both in providing an acceptable level of service, and in not demanding costly behaviours, or at least providing incentives for such behaviours. However, malicious behaviour by peer is due not only by the will to attack the system or to get higher credits, but also by the cost of verifying the authenticity of files: the user needs a correct version of the P2P software, has to find a suitable viewer, to select a not yet verified file, to wait until it opens, to check the content (sometimes an embarrassing task) and to signal the inauthenticity of the file. Since controlling the authenticity of files is costly, and this operation is a prerequisite for giving a feedback on the cooperativity of peers, the RMS is not granted to function as expected. Most approaches, most notably EigenTrust (Kamvar et al., 2003), assume that verification is made automatically upon the start of download of the file. By looking as we do at the human factor in dealing with RMS, we can question their real applicability, a question which remains unanswered in the simulation based tests made by the authors. To provide an answer to this question it is necessary to build a simulation tool aiming at a more accurate modelling of the users' behaviour rather than at modelling the RMS in detail.

3 MODEL

The P2P network is modelled as an undirected and non-reflexive graph. Each node is an agent, representing a P2P user. Agents are reactive: their behaviour is thus determined a priori, and the strategies are the result of the stimuli coming from the environment and of the condition-action rules. Their behaviour is illustrated in next section. Formally the multi agent system is defined as *MAS = <Ag; Rel>*, with *Ag* set of nodes and *Rel* set of edges. Each edge among two nodes is a link among the agents and is indicated by the tuple *< ai; aj >* with *ai* and *aj* belonging to *Ag*. Each agent features the following parameters:

– Unique ID (identifier),
– Reputation value (or credits) N(ai),
– Set of agent's neighbors RP(ai),
– Set of owned resources RO(ai),
– Set of goals (resource identifiers) RD(ai),
– Set of resources being downloaded P(ai),
– Set of pairs < supplier; resource >.

A resource is a tuple *<Name, Authenticity>*, where *Name* is the resource identifier and *Authenticity* is a Boolean attribute indicating whether the resource is authentic or not. The agent owning the resource, however, does not have access to this attribute unless he verifies the file.

The resources represent the object being shared on the P2P network. A number of resources are introduced in the system at the beginning of the simulation; they represent both the owned objects and the agents' goals. For coherence, an owned resource can't be a goal, for the same agent. The distribution of the resource is stochastic. During the simulation, other resources are stochastically introduced. In this way, each agent in the system has the same probabilities to own a resource, independently from her inner nature (malicious or loyal). In the same way also the corresponding new goals are distributed to the agents; the difference is that the distribution probability is constrained by its being possessed by an agent. Formally *R* be the set of all the resources in the system. We have that: $RD(a_i) \subseteq R, RO(a_i) \subseteq R$ and $RD(a_i) \cap RO(a_i) = \emptyset$.

Each agent in the system features a set of neighbours *N(ai)*, containing all the agents to which she is directly linked in the graph: $N(a_i) = \{a_j \in Ag \mid < ai; aj > \in Rel\}$. This information characterizes the information of each agent about the environment. The implemented protocol is a totally distributed one, so looking for the resource is heavily based on the set of neighbours.

In the real word the shared resources often have big dimensions; after finding the resource, a lot of time is usually required for the complete download. In order to simulate this the set of the "resources being downloaded" (*Ris*) introduced. These are described as *Ris = <resource ID, completion, check status>*, where *ID* is the resource identifier, *completion* is the percentage already downloaded and "*check status*" indicates whether the resource has been checked for authenticity or not. In particular, it can be not yet verified, verified and authentic and verified and inauthentic: $\text{check status} \in \{\text{NOT CHECKED; AUTH; INAUTH}\}$

Another information is *ID* of the provider of a certain resource, identified by P(ai). Each agent keeps track of those which are uploading to him, and

this information is preserved also after the download is finished. The real P2P systems allow the same resource to be download in parallel from many providers, to improve the performance and to split the bandwidth load. This simplification should not affect the aggregate result of the simulation, since the negative payoff would reach more agents instead of just one (so the case with multiple provider is a sub-case of that with a single provider).

3.1 The Reputation Model

In this work we assume a simple idealized model of reputation, since the objective is not to prove the effectiveness of a particular reputation algorithm but to study the effect of users' behaviour on a reputation system. We use a centralized system which assumes the correctness of information provided by users, e.g., it is not possible to give an evaluation of a user with whom there was no interaction. The reason is that we focus on the behaviour of common agents and not on hackers who attack the system by manipulating the code of the peer application. In the system there are two reputation thresholds: the first and higher one, under which it's impossible to ask for resources to other agents, the second, lower than the other, which makes it impossible even to share the owned files. This guarantees that an agents that falls under the first one (because she shared too many inauthentic files), can still regain credits by sharing authentic ones and come back over the first threshold. On the contrary, if she continues sharing inauthentic files, she will fall also under the second threshold, being de facto excluded from the network, still being a working link from and to other agents.

3.2 The User Model

Peers are reactive agents replying to requests, performing requests or verifying files. While upload is performed each time another agent makes a request, requesting a file and verification are performed (in alternative) when it is the turn of the agent in the simulation. All agents belong to two disjoint classes: malicious agents and loyal agents. The classes have different behaviours concerning uploading, while they have the same behaviour concerning downloading and verification: malicious agents are just common agents who exploit for selfishness the weaknesses of the system. When it is the turn of another peer, and he requests a file to the agent, he has to decide whether to comply with the request and to decide how to comply with it.

- The decision to upload a file is based on the reputation of the requester: if it is below the "replying threshold", the requestee denies the upload (even if the requestee is a malicious agent).

- The "replyTo" method refers to the reply each agent gives when asked for a resource. When the agent is faced with a request he cannot comply but the requester's reputation is above the "replying threshold", if he belongs to the malicious class, he has to decide whether to create and upload an inauthentic file by copying and renaming one of his other resources. The decision is based depending on a parameter. If the resource is owned, she sends it to the requesting agent, after verifying if her reputation is higher than the "replying threshold". Each agent performs at each round of simulation two steps:

1) Performing the downloadings in progress. For each resource being downloaded, the agents check if the download is finished. If not, the system checks if the resource is still present in the provider's "sharing pool". In case it's no longer there, the download is stopped and is removed from the list of the "owned resources". Each file is formed by n units; when 2/n of the file has been downloaded, then the file gets automatically owned and shared also by the agent that is downloading it.

2) Making new requests to other peers or verifying the authenticity of a file downloaded or in downloading, but not both:

a) When searching for a resource all the agents within a depth of 3 from the requesting one are considered. The list is ordered by reputation. A method is invoked on every agent with a reputation higher than the "requests threshold", until the resource is found or the list reaches the ending point. If the resource is found, it's put in the "downloading list", the goal is cancelled, the supplier is recorded and linked with that specific download in progress and her reputation is increased according to the value defined in the simulation parameters. If no resource is found, the goal is given up.

b) Verification means that a file is previewed and if the content does not correspond to its description or filename, this fact is notified to the reputation system. Verification phase requires that at least one file must be in progress and it must be beyond the 2/n threshold described above. An agent has a given probability to verify instead of looking for a new file. In case the agent verifies, a random resource is selected among those "in download" and not checked. If authentic, the turn is over. Otherwise, a "punishment" method

is invoked, the resource deleted from the "downloading" and from the "owned " lists and put among the "goals" once again.

The RMS is based on the "punishment" method which lowers the supplier's reputation, deletes her from the "providers" list in order to avoid cyclic punishment chains, and recursively invokes the "punishment" method on the punished provider. A punishment chain is thus created, reaching the creator of the inauthentic file, and all the aware or unaware agents that contributed in spreading it.

4 EXPERIMENTAL RESULTS

4.1 Variables and Methodology

The simulation goes on until at least one goal exists and/or a download is still in progress. A "come-back" mode is implemented and selectable before the simulation starts, in order to simulate the real behaviour of some P2P users who, realizing that they cannot download anymore (since they have low credits or, in this case, bad reputation), disconnect their client, and then connect again, so to start from the initial pool of credits/reputation. When this mode is active, at the beginning of each turn all the agents that are under a given threshold reset it to the initial value, metaphorically representing the disconnection and reconnection. This will be discussed later.

In the following table a summary of the most important parameters for the experiments are given:

Table 1: The main parameters.

Parameters	Value
Total number of agents	50
Total number of graph edges	80
Initial reputation (credits) for each agent	50
Percentage of loyal agents	5
Total number of resources at the beginning	50
Threshold to share owned resources	25
Threshold for requesting resources	10
Number of turns for introduction of new resources	1
Number of new resources to introduce	3
Total number of steps	2000

In all the experiments, the other relevant parameters are fixed, while the following ones change:

Table 2: The scenarios.

Parameters	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6
positive payoff	1	1	1	1	1	1
negative payoff	3	3	4	8	0	2
verification percentage	30%	40%	30%	30%	30%	40%

A crucial index, defining the wellbeing of the P2P system, is the ratio among the number of inauthentic resources and the total number of files on the network. The total number is increasing more and more over time, since new resources are introduced iteratively. Another measure collected is the average reputation of loyal and malicious agents at the end of the simulation; in an ideal world, we expect malicious ones to be penalized for their behaviour, and loyal ones to be rewarded. The results were obtained by a batch execution mode for the simulation. This executes 50 times the simulation with the same parameters, sampling the inauthentic/total ratio every 50 steps. This is to overcome the sampling effect; many variables in the simulation are stochastic, so this technique gives an high level of confidence for the produced results. In 2000 turns, we have a total of 40 samples. After all the executions are over, the average for each time step is calculated, and represented in a chart. In the same way, the grand average of the average reputations for loyal and malicious agents is calculated, and represented in a bar chart. In figure 1, the chart with the trend of inauthentic/total resources is represented for the results coming from experiments 1, 2, 3, 5 and 6. The results of experiment 4 is discussed later.

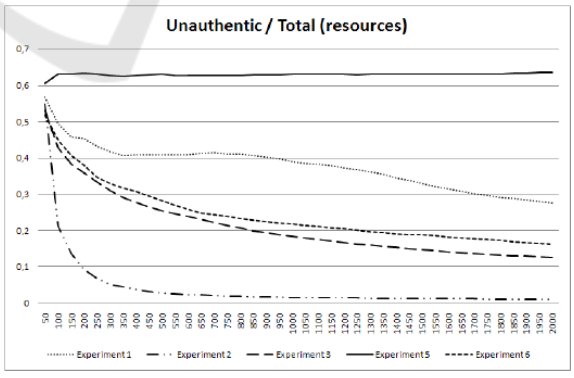

Figure 1: The trends of the results.

4.2 Evaluation of Results

Experiment 5 depicts the worst case: no negative payoff is given. The ratio initially grows and, at a certain point, it gets constant over time, since new

resources are stochastically distributed among all the agents with the same probability. In this way also malicious agents have new resources to share, and they will send out inauthentic files only for those resources they do not own. In the idealized world modelled in this simulation, since agents are 50 malicious and 50 loyal, and since the ones with higher reputation are preferred when asking for a file, it's straightforward that malicious agents' reputation fly away, and that an high percentage of files in the system are inauthentic (about 63%). Experiment 1 shows how a simple RMS, with quite a light punishing factor (3) is already sufficient to lower the percentage of inauthentic files in the network over time. We can see a positive trend, reaching about 28% after 2000 time steps, which is an over 100% improvement compared to the situation in which there was no punishment for inauthentic files. In this experiment the verification percentage is at 30%. This is quite low, since it means that 70% of the files remain unchecked forever (downloaded, but never used). In order to show how much the human factor can influence the way in which a RMS works, in experiment 2 the verification percentage has been increased up to 40%, leaving the negative payoff still at 3. The result is surprisingly good: the inauthentic/total ratio is dramatically lowered after few turns (less than 10% after 200), reaching less than 1% after 2000 steps. Since 40% of files checked is quite a realistic percentage for a P2P user, this empirically proves that even the simple RMS proposed here dramatically helps in reducing the number of inauthentic files. In order to assign a quantitative weight to the human factor, in experiment 3, the negative payoff is moved from 3 to 4, while bringing back the verification percentage to 30%. Even with a higher punishing factor, the ratio is worse than in experiment 2, meaning that it's preferable to have a higher verification rate, compared to a higher negative payoff. Experiment 6 shows the opposite trend: the negative payoff is lighter (2), but the verification rate is again at 40%, as in experiment 2. The trend is very similar – just a bit worse - to that of experiment 3. In particular, the ratio of inauthentic files, after 2000 turns, is about 16%. At this point, it gets quite interesting to find the "break even point" among the punishing factor and the verification rate. After some empirical simulations, we have that, compared with 40% of verification and 3 negative payoff, if now verification is just at 30%, the negative payoff must be set to a whopping value of 8, in order to get a comparable trend in the ratio. This is done in experiment 4: after 2000 turns, there's 1% of inauthentic files with a negative payoff of 3 and a verification percentage of 40%, and about 0.7 with 8 and 30% respectively. This clearly indicates that human factor (the files verification) is crucial for a RMS to work correctly and give the desired aggregate results (few inauthentic files over a P2P network). In particular, a slightly higher verification rate (from 30% to 40%) weights about the same of a heavy upgrade of the punishing factor (from 3 to 8).

Besides considering the ratio of inauthentic files moving on a P2P network, it's also crucial to verify that the proposed RMS algorithm could punish the agents that maliciously share inauthentic files, without involving too much unwilling accomplices, which are loyal users that unconsciously spread the files created by the former ones. In the agent based simulation, this can be considered by looking at the average reputation of the agents, at the end of the 2000 time steps. In the worst case scenario, the malicious agents, that are not punished for producing inauthentic files, always upload the file they are asked for (be it authentic or not). In this way, they soon gain credits, topping the loyal ones. Since in the model the users with a higher reputation are preferred when asking files, this phenomenon soon triggers an explosive effects: loyal agents are marginalized, and never get asked for files. This results in a very low average reputation for loyal agents (around 70 after 2000 turns) and a very high average value for malicious agents (more than 2800) at the same time. In experiment 1 the basic RMS presented here, changes this result; even with a low negative payoff (3) the average reputations after 2000 turns, the results are clear: about 700 for loyal agents and slightly more than 200 for malicious ones. The algorithm preserves loyal agents, while punishing malicious ones. In experiment 2, with a higher verification percentage (human factor), we see a tremendous improvement for the effectiveness of the RMS algorithm. The average reputation for loyal agents, after 2000 steps, reaches almost 1400, while all the malicious agents go under the lower threshold (they can't either download or share resources), with an average reputation of less than 9 points. Experiment 3 explores the scenario in which the users just check 30% of the files they download, but the negative payoff is raised from 3 to 4. The final figure about average reputations is again very good. Loyal agents, after 2000 steps, averagely reach a reputation of over 1200, while malicious ones stay down at about 40. This again proves the proposed RMS system to be quite effective, though, with a low verification rate, not all the malicious

agents get under the lower threshold, even if the negative payoff is 4. In experiment 6 the verification percentage is again at the more realistic 40%, while negative payoff is reduced to 2. Even with this low negative payoff, the results are good: most malicious agents fall under the lowest threshold, so they can't share files and they get an average reputation of about 100. Loyal agents behave very well and reach an average reputation of more than 900. Experiment 4 is the one in which we wanted to harshly penalize inauthentic file sharing (negative payoff is set at 8), while leaving an high laxity in the verification percentage (30%). Unlikely what it could have been expected, this setup does not punish too much loyal agents that, unwillingly, spread unchecked inauthentic files. After 2000 turns, all the malicious agents fall under the lowest threshold, and feature an average reputation of less than 7 points, while loyal agents fly at an average of almost 1300 points. The fact that no loyal agent falls under the "point of no return" (the lowest threshold) is probably due to the fact that they do not systematically share inauthentic files, while malicious agents do. Loyal ones just share the inauthentic resources they never check. Malicious agents, on the other side, always send out inauthentic files when asked for a resource they do not own, thus being hardly punished by the RMS, when the negative payoff is more than 3.

4.3 Comeback Mode

It is straightforward to imagine that an agent could simply disconnect and reconnect to the network, in order to start from an initial reputation value and overcome the limits of this RMS. In order to simulate this, a "comeback mode" (CBM) has been implemented in the simulation. The agents, when reaching the lowest threshold, are simply reset to the initial reputation value (that's what would happen if they disconnected and reconnected).

Even with CBM activated, the results are very similar to those in which this mode is off. They are actually a bit worse when the negative payoff is low (3) and so is the verification percentage (30%): the ratio of inauthentic files in the network is quite high, at about 41% after 2000 turns versus the 27% observed in experiment 1, which had the same parameters, but no CBM. When the verification percentage is increased to 40%, though, things get quite better. Now the ratio of inauthentic files has the same levels as in experiment 2 (less than 1% after 2000 steps). Also with a lower verification percentage (again at 30%), but leaving the negative payoff at 4, the figure is almost identical to the one with the same parameters, but without a CBM. After 2000 turns, the inauthentic files ratio is about 12%.

The experiments show that malicious agents, even resetting their own reputation after going below the lowest threshold, can't overcome this basic RMS, if they always produce inauthentic files. This happens because, even if they reset their reputation to the initial value, it's still low compared to the one reached by loyal agents; if they shared authentic files, this value would go up in few turns, but since they again start spreading inauthentic files, they almost immediately fall under the thresholds again.

4.4 Scaling Issue

Changing the numbers of agents, agent based models can suffer from scaling issues, meaning that the results obtained with a certain number of agents don't hold when this number changes significantly. To verify this, two more experiments were carried on. The number of agents is increased to 150 (three times the previous value). Coherently, the number of edges, the initial pool of resources and the number of resources introduced at each turn are also tripled. The trend is very similar: with a low negative payoff (3) and a low verification rate (30%), we have just a slightly higher ratio of inauthentic files. The same comparison is carried on with the average final reputations. Again, the results are very similar, even if the system with more agents has a slightly worse aggregate behaviour than the smaller one. Generally speaking we conclude that the difference is very low, so the scaling issue is not influencing the results shown, on a 1:3 basis. In future works this study will be extended to even more agents.

5 CONCLUSIONS AND OUTLOOK

First of all in this paper we look for the factors controlled by users determining the effectiveness of RMSs in P2P network. Two are the critical points: the decision of sharing inauthentic files and the decision to verify or not the downloaded files.

While the benefit of not verifying is determined by the saved time (verifying is incompatible with making new searches and starting new downloads), the benefit of spreading inauthentic files must be confirmed by the simulation. Without the mechanism for punishing malicious agents, inauthentic files increase sharply, and at the end of simulation the reputation of malicious agents is

much higher than loyal ones', and this is reached very soon. Thus, producing inauthentic files is a worthy strategy, under no enforcement mechanism.

However, the behaviour of malicious agents strikes back against them: they are not attackers and have the goal of downloading resources, as well.

Here we assume that if a file is verified, then the reputation of the uploader is decreased immediately, due to the lower cost of this action. A more fine grained model should consider also this human factor. Analogously we do not consider the possibility to punish peers without first receiving and checking a file - a behaviour which should be prevented by the software itself - as well as we do not consider the possibility of punishing even if the file is authentic. As stated in the Introduction, our goal is to model the behaviour of normal user, not of hackers attacking the system.

The second question of the work is: how to evaluate the role of these factors by means of agent based simulation? Which factors have most impact?

The simulation framework for reputation gives interesting results: the key factor to lower the number of inauthentic files in a file sharing P2P system is the proportion of verifications made by peer. Even a low figure like 30% sharply limits the behaviour of malicious agents when we do not consider the possibility of whitewashing after comeback. The role of punishment, in terms of reputation points, has instead a more limited impact.

Surprisingly, even when whitewashing is allowed, the number of inauthentic files in the system can be limited if peers verify files 40% of the times. The same results cannot be achieved by increasing the figure of the punishment and decreasing the proportion of verifications.

The moral of our study is that a mechanism for stimulating users to check the authenticity of files should be promoted, otherwise the entire file sharing system is flooded by inauthentic files. In contrast, usual approaches to RMS (Josang et al., 2007) consider verification as automatic, thus ignoring the human factor: since we show that verification has a sharp effect according to its rate, it cannot be ignored in simulating the effect of a RMS. Thus, we identify the conditions when even a simple RMS can dramatically reduce the number of inauthentic files over a P2P system and penalize malicious users, without banishing them from the network, as proposed in other models based on ostracism. The proposed model is very simple. The reader must be aware of several limitations, which are the object of ongoing work. Resources are not divided in categories; inauthentic files, in reality, are mostly found in newer resources. Thus, we are aiming at using real data to differentiate the kinds of resources, distinguishing in particular newly requested resources. At present we distinguish malicious agents from loyal agents based on their response, but all the agents have the same behaviour for other aspects, e.g. they verify with the same proportion. It could be useful to simulate using different parameters in each agent of the two classes.

Bandwidth is not considered: thus all downloads proceeds at the same rate, even if the decision to upload to a peer is based on his reputation. Moreover, a peer decides to upload on the basis of which agent has the highest reputation. It is well known that this algorithm can create unbalance among peers, but we abstract here from this problem, since we are not proposing a new P2P mechanism but checking the efficacy of a RMS on the specific problem of inauthentic files. Note, however, that this strategy has a negative effect when malicious peers get high reputation, but if the reputation system is well tuned, malicious agents should never get high reputation.

Finally, we allow agents to disconnect and reconnect, but this whitewashing happens without changing their position on the graph, or behaviour.

The real improvement in our ongoing work, however, is moving from reactive agents to more sophisticated ones, able to learn from what is happening in the network. While in the current model agents stochastically decide whether to upload an inauthentic file or not, or to verify or not, cognitive agents adapt to the circumstances, looking how many objectives they can achieve using their current strategy, and looking for new alternatives. Modelling adaptive agents allows to check for further vulnerabilities, like what happens when agents produce inauthentic files at a variable rate which does not decrease too much their reputation.

REFERENCES

Josang A., Ismail R, and Boyd. C., 2007. A survey of trust and reputation systems for online service provision. *Decis. Support Syst., 43(2):* pp 618–644.

Kamvar S. D., Schlosser M. T., and Garcia-Molina H., 2003. The eigentrust algorithm for reputation management in p2p networks. *In WWW '03: Proceedings of the 12th international conference on World Wide Web*, pp 640–651, USA, ACM Press.