Defending the Sybil Attack in P2P Networks: Taxonomy, Challenges, and a Proposal for Self-Registration

Jochen Dinger and Hannes Hartenstein
Institut für Telematik, Universität Karlsruhe (TH), Germany
dinger@tm.uka.de, hartenstein@rz.uni-karlsruhe.de

Abstract

The robustness of Peer-to-Peer (P2P) networks, in particular of DHT-based overlay networks, suffers significantly when a Sybil attack is performed. We tackle the issue of Sybil attacks from two sides. First, we clarify, analyze, and classify the P2P identifier assignment process. By clearly separating network participants from network nodes, two challenges of P2P networks under a Sybil attack become obvious: i) *stability over time, and* ii) *identity differentiation. Second, as a starting point for a quantitative analysis of time-stability of P2P networks under Sybil attacks and under some assumptions with respect to identity differentiation, we propose an identity registration procedure called* self-registration *that makes use of the inherent distribution mechanisms of a P2P network.*

1 Introduction

Peer-to-Peer (P2P) networks are logical or *virtual* networks on top of already existing networks, i.e., they are mostly built upon an underlying network like the Internet. Most P2P networks use their own 'addressing scheme' based on logical identifiers for structuring and organizing the network. In particular, structured DHT-based networks like Chord [15] and Pastry [12] use their own identifier scheme. With respect to the identifier assignment procedure various issues have to be considered: a standard requirement says that node identifiers should be globally unique and uniformly distributed over the identifier space.

When an entity can run a large number of nodes and obtain a large number of node identifiers, the P2P network can be dominated by this entity. This dominance can be used to undermine replication mechanisms, which results in less robustness. Finally one entity can control the whole P2P network. This is commonly known as Sybil attack [5]. Any design of an assignment process of node identifiers has to take this type of attack into account: the identifier assignment should therefore be verifiable and withstand a Sybil attack.

The Sybil attack is not only relevant in P2P networks, also other types of networks like ad-hoc and sensor networks [11] have to cope with this problem. However, the effect on P2P might be the most significant one because of the P2P network's 'virtual nature' (compared to some physical constraints in ad-hoc networks) and their global size.

While Douceur's seminal paper [5] has shown the restrictive fundamental constraints for a Sybil-proof system, we re-visit the design space, provide a taxonomy, and point to key observations, namely time-stability and identity differentiation. For a relaxed set of constraints we then propose a 'self-registration' mechanism that we use as a starting point for our time-stability analysis. The self-registration approach is inspired by the 'self-contained' mechanism of P-Grid [1]. Although Douceur shows that distributed approaches will never be completely Sybil-proof, we like to quantify the Sybil resistance of such distributed approaches by probabilistic analysis.

The paper is organized as follows. In Section 2 we formally define the elements of an *ID*[1] assignment procedure as a basis for our discussion. In Section 3 we survey the most relevant work. In Section 4 we classify various approaches with respect to their organizational form. In Section 5 the impact of multiple *IDs* per participants leads us to two key challenges: stability over time and differentiation between identities. We propose a P2P-based identity registration procedure in Section 6 where we also provide results via analysis and simulation. Section 7 concludes the paper.

2 Terminology and notation

First, we introduce the terminology and notation used in the rest of the paper:

- A node n_i is the atomic entity of a P2P network. All nodes of the P2P network together form the set $N = \{n_1, ..., n_n\}$ with $|N| = n$ number of involved nodes.

[1] *ID* is used as synonym to identifier

- Nodes are the atomic unit on the P2P level but not necessarily on the 'community' level above. We call the entity on the level above participant p_i. One participant can control one or more nodes. All participants together form the set $P = \{p_1, ..., p_m\}$ with $|P| = m$ number of involved participants. We only take active participants into account, which means that at least one node belongs to each participant, $n \geq m$.

- The node identifier of node n_i is denoted as id_i. The identifiers are selected from an identifier space K, i.e., $id_i \in K$, $\forall i$. When unique node identifiers are assigned in a P2P network, then we have: $\forall n_i, n_j \in N \; with \; i \neq j : id_i \neq id_j$.

- Now we can also define participant specific node sets. N_i is the set of nodes which belong to participant p_i. We assume that a node always belongs to exactly one participant, thus: $\forall N_i \cap N_j = \emptyset \; with \; i \neq j$ and $N_1 \cup \ldots \cup N_m = N$.

- Besides P2P specific identifiers, nodes can also have external identifiers. The external *ID* of a node n_i is denoted as eid_i. For instance, an IP address can be such an external identifier. One node can also have multiple external *IDs* like IP address and PGP certificate.

- Additionally, we define a constant a that denotes the maximum of allowed nodes per participant. Hence, we have $\forall p_i \in P : \; |N_i| \leq a$. Only when a Sybil attack was performed successfully, $\exists p_i \in P : \; |N_i| > a$.

- Furthermore, we define $hash(i)$ as a strong one-way hash function like SHA-256.

3 Related work

The Sybil attack in P2P networks was first mentioned in [5]. Douceur shows that, if a single malicious entity can present multiple identities this entity can control the whole network. He argues that under realistic assumptions of resource distribution and coordination only a central organized authority can prevent from a Sybil attack. But he says that implicit identification authorities like ICANN can be sufficient for Sybil resistance if they are 'mindfully' used.

In [3], Cheng and Friedman show that symmetric reputation functions cannot be resistant to Sybil attacks. Because an attacker can always create a copy of an existing network and then adjust its reputation arbitrary. 'Sybilproof reputation mechanism' have to use asymmetric reputation function that means some trust to dedicated nodes is necessary.

Sit and Morris [13] presented a categorization of P2P security issues. In their design principles they argue that the *ID* assignment has to be done in a verifiable way. They also argue that the assignment process should be secure such that a node cannot arbitrarily choose its node identifier. Additionally, they mention that a central identification authority is not desirable in all situation.

Castro et al. [2] argue that it is fundamental for P2P networks that the uniform distribution of node identifiers cannot be controlled by an attacker. They propose a trusted central authority as solution, like Pastry [12] uses. They explicitly allow multiple node *IDs* per IP address.

In [14], Srivatsa and Liu define an 'ID Mapping Scheme' in their formal model based on external identifiers. They argue that the introduction of IPv6 would be a problem to standard mappings that are just based on the IP address.

Fiat et al. [7] present S-Chord as variant of Chord. They tackle the problem that nodes could choose *IDs* such that they are at critical positions in the Chord ring. They use a set of nodes called swarm to cope with the problem. In their modified join protocol node *IDs* are not mapped from external *IDs*, they are chosen randomly by a responsible swarm. The described S-Chord is robust under the assumption that less than $1/4 \cdot n$ nodes join the network in a period of time. In contrast to our work they limit the number of malicious nodes and not the number of malicious participants.

Danezis et al. [4] present modified routing strategies for Chord to minimize the costs induced by a Sybil attack. The routing strategies are build upon a so called bootstrap-graph, which is derived from relationships between participants that have been established outside the P2P network. Additionally trust metrics are used to minimize the probability that a malicious node is on the routing path.

A kind of self-registration mechanism is also included in P-Grid [1]. It is referred to as 'self-contained'. In difference to our approach their approach is not directed to limit Sybil attacks, instead they use it to overcome problems resulting from DHCP-based IP address assignment.

4 *ID* assignment: a taxonomy

Identifiers for nodes in P2P networks can be assigned in different ways, which will be outlined in Section 4.1. The assignment mechanism is the basis for defending against the Sybil attack. These mechanisms also have to be verifiable for other P2P nodes and node *IDs* have to be limited. This assumptions lead to three different processes:

1. During the *assignment* procedure participants obtain identifiers for their nodes.
2. *Verification* allows other nodes to distinguish between valid and counterfeit node *IDs* (as defined in [13]).
3. A *limitation* of node identifiers per participant is necessary for effectively preventing a Sybil attack.

Verification and limitation possibilities depend on the assignment method. Hence, they are discussed with the corresponding assignment method.

4.1 Classes of assignment mechanisms

We discuss assignment mechanisms under the following criteria: *i) Limitation and verification possibilities* are essential to design *Sybil resistant* P2P systems. *ii)* The *entrance barrier* in a P2P network depends on how difficult it is to obtain a new *ID*. Especially in open systems like electronic market platforms the entrance barrier should be low to encourage new participants. On the other hand the market place should be secure to encourage new participants, which means that *ID* assignment has to be done securely. *iii)* When additional resources are necessary for the assignment of identifiers, *additional costs* arise. Also questions about the *organization and control* over such entities will be raised, e. g. , when assignment entities are controlled by a major market player, competitors could be excluded or obstructed. *iv)* The *robustness* of the *ID* assignment against failures, in the sense of safety, is also an important issue.

4.1.1 Centralized identifier assignment

In this case the identifiers are assigned by a central entity like in [12]. This central entity might not be part of the P2P network. For example, such an entity could be a certification authority (CA) that assigns *IDs* and signs certification requests. This entity is central or hierarchical with regard to the organization. Hierarchical organization means that the assignment can be distributed like CA and Sub-CA, but there is still one central anchor point.

As outlined in [5] central assignment can be Sybil-proof. For example, Sybils could be prevented via additional identification processes like identity cards. Another possibility would be a fee based solution such that each participant has to pay for the assignment of a node *ID*. This offers (through increasing the fee) a tunable mechanism to defend the Sybil attack and the fees can be used for operating the central entity. On the one hand this cuts down the chance of a successful Sybil attack, but on the other hand the entrance barrier into the P2P network is raised. Centralized identifier assignment has to be operated by someone and requires technical resources, which result in additional costs and the 'organizational questions'. A central entity can result in a single-point of failure. Whereas the central entity is not essential for the function of the P2P network itself, it is vital for nodes trying to join the P2P network.

When *IDs* are assigned by a central entity, the central entity has to offer a verification method. A simple solution would be the typical CA scenario. Each node knows the public key of the central entity and the central entity signs each node identifier with its private key. Hence, each node can prove if a node *ID* is valid or not. The central entity has 'global' knowledge. Thus, the number of allowed nodes per participant can be limited easily.

4.1.2 Distributed assignment with external identifiers

Identifiers can also be derived from existing external identifiers eid_i, because centralized and direct assignment of identifiers is not always possible or wanted [13]. For mapping the external *IDs* to P2P *IDs* mapping functions are necessary, which will be discussed in detail in Section 5. IP addresses are a typical example for such external *IDs*. The ICANN, sub registries, and providers assign and ensure that IP addresses are globally unique.

The external *ID* based assignment can only be as secure as the assignment of the external *IDs*. If the external *ID* assignment process is vulnerable to the Sybil attack, the P2P *ID* assignment cannot be secured. The external assignment authorities can be organized in a centralized or distributed manner. Whereas in the case of distributed assignment the same problems would arise as outlined in Section 4.1.4.

The usage of external *IDs* does not require additional entities in contrast to the central *ID* assignment. Thus, no additional costs occur and no operation issues arise. In terms of the entrance barrier the mapping has the advantage that participants do not have to obtain a new separate *ID* for the P2P network. They can just use their existing eid_i and automatically map it to a P2P specific identifier. When the mapping of *IDs* is secure, new participants can be convinced by the 'security' of the external *ID* assignment.

For verification purposes the external *ID* and furthermore the mapping function have to be verifiable. For example, the mapping function of the external identifier eid_i is $hash(eid_i) = id_i$. Then, the following invariant exists: $\forall n_i \in N : hash(eid_i) - id_i = 0$.

If the external *ID* assignment is Sybil-proof and the mapping is 'secure', the P2P network is also Sybil-proof. Thus we have to ensure that the external *IDs* per participant are limited. In the following sections we will show the impact of multiple external *IDs* per participant and its impact.

4.1.3 Distributed assignment with free / unrestricted identifiers

Distributed assignment with free/unrestricted identifiers indicates that each participants can choose more or less arbitrary node identifiers for its nodes. There can be mechanisms to guarantee that these identifiers are globally unique, but there is no control about how many node *IDs* are obtained by one participant. An advantage of this mechanisms is that the assignment can be realized easily. The free assignment does not require additional entities and therefore does not generate additional costs. Hence, the entrance barrier is very low.

The free and unrestricted assignment can be verified, if resource consuming proofs (e.g., crypto puzzle [9]) are used. A node n_i has therefore to present the solution of the resource consuming task that was assigned to it. Another node n_j can then prove id_i. For example, the result of a crypto puzzle is presented. When there are no resource consuming proofs, there is no verification counterpart. Thus the node *ID* cannot be verified. The robustness with regard to safety of the free assignment is very good, because no central components are involved and all tasks can be distributed uniformly to all nodes.

The major disadvantage is inherently no limitation on node *IDs* per participant, because each participants can generate an arbitrary number of node *IDs*. Douceur [5] proves that this mechanism cannot be secure with regard to the Sybil attack. It is also shown that even crypto puzzles and similar resource consuming mechanisms cannot make this assignment mechanism resistant to Sybil attacks.

4.1.4 Distributed group-based identifier assignment

Besides a free and unrestricted assignment like in the previous section, we might also think about a group-based assignment process such that a number of nodes assert that a specific node identifier id_i is correct.

The entrance barrier is very low and no additional costs arise like in the free assignment. Because of the distributed manner the robustness with regards to safety is very good. The only way to apply verification methods is the same as mentioned in Section 4.1.3. The mechanism seems to be more secure than the free assignment, but in [5] it is also shown that this mechanism is not Sybil-proof.

4.2 Relation to trust and reputation

While trust mechanisms are not in the focus of this paper, we would like to note that even with suitable trust mechanisms, the Sybil attack is still a serious danger to P2P networks. As mentioned in [6], if it is cheap to obtain new identifiers, participants can easily white wash themselves through obtaining new identifiers. Therefore reasonable trust and reputation mechanisms rely on P2P networks that are robust against Sybil attacks. Otherwise costs like social cost increase. For example, when the reputation of a new identifier is very bad, the entrance barrier into the P2P network is much higher than compared to a neutral reputation for new *IDs*. Problems also arise through colluding groups of malicious nodes, which then can increase their rating by themselves [3].

Certainly, central entities or the external entity like the ICANN have to be trusted. But this might be reasonable because we already trust such entities today, what is an advantage of the external identifiers based assignment compared to the centralized assignment. The external *ID* based assignment can easily involve well known and established trustable parties without their direct cooperation. In contrast to the central assignment, when trustable parties have to cooperate directly with the initiators of the P2P network.

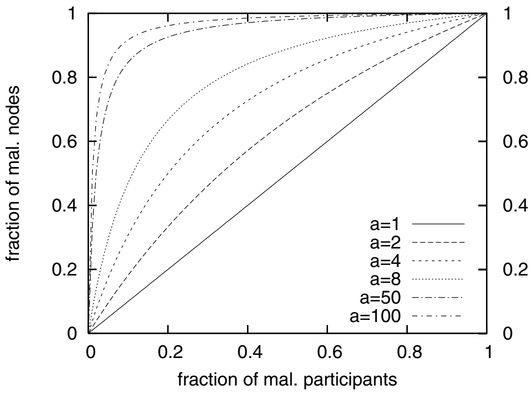

Figure 1. Ratio between malicious participants p_p and actual fraction of malicious nodes p_n

5 Mapping of node identifiers and the challenge of multiple node identifiers

In this section we focus on the mapping of external identifiers to P2P specific identifiers based on the class defined in Section 4.1.2. These mappings can be subdivided in the following mappings ($i, j \in \mathbb{N} : i, j \geq 1$): *i)* $1 \rightarrow 1$: One external *ID* is mapped to one P2P specific *ID*. *ii)* $1 \rightarrow i$: One external *ID* is mapped to a set of P2P specific *IDs*, whereas the cardinality of this set is a fixed positive integer. *iii)* $j \rightarrow 1$: Multiple external *IDs* are mapped to one P2P specific *ID*. *iv)* $j \rightarrow i$: Multiple external *IDs* are mapped to multiple P2P specific *IDs* , whereas the cardinality of P2P specific set is a fixed positive integer.

The mapping has to be verifiable. The mappings of the first and third case can be solved relatively easy through hashing the external *IDs* with a secure hash function. The 4th case can be substituted by: $j \rightarrow 1 \rightarrow i$. A solution for the 2nd case would be hashing a concatenated string[2] like $hash(i \oplus ipAddress)$, where i is a counter from 1 *to* a. But this solution requires a coordination among the nodes that rely on the same external *ID*. In Section 6 we will outline this problem on a concrete example.

5.1 Impact of multiple *IDs* per participant

In the following, the impact of allowing multiple identifiers per participants ($a > 1$) will be outlined. If we set $a > 1$, participants can 'legally' obtain more than one node *ID* per participant. We as well as other authors like [5, 2] assume that $a > 1$ is a reasonable assumption for various identifier assignment procedures.

However, this assumption may have a huge impact on the fraction of malicious nodes. We define a malicious partici-

[2] string concatenation is denoted by $\oplus$

pant as a participant that tries to obtain as much node *IDs* as possible and each well-behaving node just obtains one node *ID*. The ratio between the fraction of malicious participants p_p and malicious nodes p_n is calculated as follows (n_{mal} denotes the number of malicious nodes):

$$p_n = \frac{n_{mal}}{n} = \frac{a * p_p}{(1 - p_p) + a * p_p}$$

As shown in the Figure 1, the actual fraction of potential malicious nodes rapidly increase even for small a. Assume the number of nodes per participants is $a = 8$ and 2% of all participants are malicious, could result in over 14% malicious nodes, which is immense. For realistic results it is essential to start from the fraction of participants and then estimate a correctly. One can argue that all nodes have to collude, but this is reasonable due to the fact that they can also collude passively by not contributing resources to the P2P system or not adhering to the rules.

5.2 Time-stability and identity differentiation

P2P networks are not static since nodes join and leave the network all the time. Even when we assume that the P2P network is free of Sybils at one moment in time, this can change easily. Sybils can then achieve a critical mass and undermine the network. Centralized mechanism are not necessarily affected by this problem since they always rely on one central trustable entity and not on a majority decision of other nodes. Each distributed approach faces this time-stability problem. Hence, distributed approaches have to ensure that malicious nodes never reach a critical mass. In Figure 1 we can observe the underlying non-linearity. Thus, the key question for a distributed identifier assignment procedure is not whether it is definitely Sybil-proof (which is not, see results by Douceur), but for what time period the network can be assumed to be not dominated by Sybils with high probability.

Identity differentiation has two faces: first, we have to differentiate participant identifiers. As we have seen in Section 5.1 small increases of participants can result in huge numbers of additional malicious nodes. Centralized as well as distributed approaches face this problem because the distinction of identities has to be done by both. Second, node identifiers have to be differentiated. Hence, the assignment has to be verifiable.

6 Self-registration

With the insights of the previous sections in mind, we designed an identifier registration service that is based on a distributed identifier assignment using external identifiers. The main idea is to use the P2P network itself as registration entity, therefore, we call this approach 'self-registration'. Clearly, this approach is not Sybil-proof, but we see a potential that this approach can be Sybil resistant for an amount of time with high probability. This approach represents a starting point to study the time-stability of P2P networks without central entity with respect to Sybil attacks.

Essentially, the approach works as follows: nodes calculate their identifier based on their IP address and also take the port of the connection into account. The node then registers its *ID* in the P2P network at already successfully registered nodes. The registration is only based on the IP address or parts of the IP address. This will be discussed in detail later. After a successful registration a new node can join the network. Other nodes will verify its registration when they try to integrate the node in the P2P network.

We focus on this approach because of the advantages mentioned in Section 4.1. In particular, we think that the entrance barrier should be as low as possible and the identifier assignment should be distributed because of the decentralized manner of a P2P network.

The 'standard' approach (like in [15]) of mapping IP addresses to P2P *IDs* is to use a hash function $hash(eid_i) = id_i$. This approach has two disadvantages [2, 14]. First, this approach cannot work for Network Address Translation (NAT), because these networks mostly have only one public IP address. Thus, only one node can reside behind the NAT gateway. Second, in IPv6 nodes can obtain a huge number of IP addresses, e. g. , through using the privacy extensions defined in [10]. We do not argue that an IP address is a secure identifier that cannot be counterfeit. But considering globally-routable IP addresses, (parts of the) addresses are more or less restricted per participant. In IPv4 ordinary users only get one or a small amount of IP addresses from their provider. In IPv6 users can get a nearly arbitrary number of addresses. However, we argue that one user only has one or a small amount of different IPv6 address prefixes because of routing issues. So, an idea would be to take only the first 64 bit of the IPv6 address. We allow multiple node *IDs* for one IPv4 address and in the case of IPv6 we use only the prefix of the address. We denote the IP address (IPv4) or prefix of the IP address (IPv6) as $ipAddressPre$. It is reasonable to analyze the usage of IP addresses as external *IDs* because IP addresses are the only globally available external *IDs* in the Internet.

6.1 Byzantine failure

A P2P network has also to cope with the Byzantine Generals Problem [8]. In short, the Byzantine Generals Problem is as follows. A decision has to be made by one participant, whereby the decision basis comes from well-behaving *and* malicious participants and nodes respectively. Taking multiple nodes into account, one can try to come to the correct decision even with some malicious nodes involved.

The Byzantine problem is not relevant for centralized identifier assignment because the verification can be done directly. But we have to cope with the problem because our approach is distributed and uses indirect verification. Thus, a node has always to be registered at r different nodes. We call this replication factor r. For a successful registration at least $\lceil \frac{r}{2} \rceil$ nodes have to confirm that the node is registered correctly. Not all nodes have to confirm the registration because then a reverse attack would be possible: legitimate nodes could be excluded by denying their correct registration. We assume that initially a number of nodes is well-behaving. This is rational since we assume that the initiators of a P2P network are not malicious.

6.2 Algorithm

Our approach is based on the Chord [15] protocol and should be easily applicable for other DHT protocols as well. The self-registration procedure requires to modify the join phase and stabilize phase of Chord. The join phase is modified as follows:

1. A new node n_i calculates its Chord *ID* id_i based on its IP address and port through a hashing function: $eid_i = ipAddress_i \oplus port_i$, $hash(eid_i) = id_i$.
2. After node n_i has calculated its node *ID* id_i, the node has to register id_i at r registration nodes. These registration node *IDs* are computed as follows[3]: $regId_i^j = hash(j \oplus ipAddressPre_i)$ $(1 \leq j \leq r)$. Note that only the IP address (IPv4) or the prefix of the address (IPv6) is used, i.e., the registration is participant-based.
3. Finally, the node n_i tries to join the network. Therefore node n_i informs its successor and will be integrated in the network through the recurring stabilizing process.

When a registration request reaches a responsible registration node, this node first checks if the actual number of registered nodes for this IP address (prefix) $ipAddressPre_i$ is less than a. If this is the case, the node stores the node identifier (id_i), the IP address and port (eid_i). Thus, each node stores a list of IP addresses. For each IP address (prefix) $ipAddressPre$ a node stores a list of actual registered nodes. The stabilizing phase is modified as follows:

1. When a join request reaches a node n_j from a new node n_i, the node n_j will first verify the id_i of this new node n_i. This is done by the invariant $hash(eid_i) - id_i = 0$ and by issuing a ping request to verify the eid_i.
2. Afterwards, the node n_j will calculate the appropriate registration identifiers $regId_i^j$ of the new node n_i in the same manner as in the join process. Then the node n_j checks the correct registration of the new node n_i by asking the responsible nodes.
3. If the number of positive replies is $\geq \lceil \frac{r}{2} \rceil$, the new node n_i will be accepted. Thus the predecessor entry will be adjusted. We denote the number of positive replies on registration verification as k.

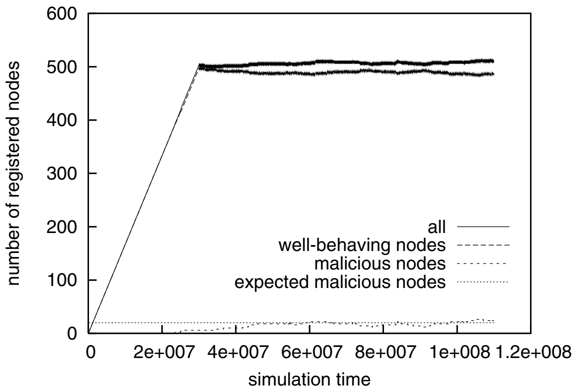

Figure 2. Simulation run that shows a prevented Sybil attack

If a new node joins successfully, that node is also integrated in the registration process and therefore registration data has to be moved from other nodes according to the Chord protocol. The same happens when a node leaves the network that registration data also has to be moved.

6.3 Evaluation

For the simulation we implemented Chord as well as the extension described above in J-Sim [16]. The Chord ring size was 2^{64} for all simulations. Our setup was as follows: nodes try to join the network and leave it after some period of time. Malicious participants first register the allowed number a of nodes and then try to register more nodes. Such 'illegal' registrations are tried periodically by malicious participants. Furthermore, malicious nodes allow to register an arbitrary number of nodes. Thus, they try to undermine our limitation method. Because of that, a malicious node may register more than the allowed number of nodes when the parameters are not adjusted correctly.

Figure 2 shows a simulation run with the parameters $p_p = 0.02$, $a = 2$, and $r = 5$. This result shows that the number of malicious nodes is always around the expected number of malicious nodes and never exceeds it significantly. Thus, malicious nodes and participants respectively have not been able to launch a successful Sybil attack. During the simulation a new participant joins the network every 60,000 time units (simulation time). The stabilize process is

[3] Again, string concatenation is denoted by $\oplus$

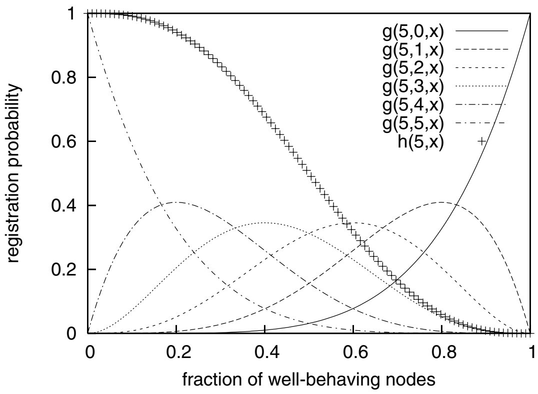

Figure 3. Probabilities of a 'false registration' in comparison with the fraction of well-behaving nodes

run every 14,000 time units. After $30 * 10^6$ time units nodes leave the network. When a new participant is created it is chosen to be good or malicious with probability $(1-p_p)$ and p_p respectively. After creation of a malicious participant, the participant tries to create as many nodes as possible. The number of expected malicious nodes is calculated according to Section 5.1: $p_n \approx 0.0392 \Rightarrow n_{malicious} = m * p_n \approx 20$ ($m = \frac{lifetime}{creationInterval} = \frac{30*10^6}{60{,}000} = 500$).

6.3.1 Effectiveness

For the following results we set a to one. For $a > 1$ we would have to combine these results with the results discussed in Section 5.1. Hence, we have $p_p = p_n$.

If a node is malicious, it will confirm also 'illegal' registrations. We denote the number of malicious nodes involved in the registration process as k, which means that k of the r registration nodes are malicious and claim a correct registration that is incorrect. According to our registration algorithm $\geq \lceil \frac{r}{2} \rceil$ nodes have to confirm that a registration is correct. If a malicious participant is able to register more than the allowed number a of nodes per participant, i.e. $k \geq \lceil \frac{r}{2} \rceil$, we refer to it as 'false registration'.

The probabilities that k nodes confirm a registration are calculated by the function $g(r, k, p_n)$. The probability of a 'false registration' corresponds to $h(r, p_n)$.

$$g(r, k, p_n) = (1 - p_n)^{r-k} * p_n^k * \binom{r}{k}$$

$$h(r, p_n) = \sum_{i=\lceil \frac{r}{2} \rceil}^{r} g(r, i, p_n)$$

Figure 3 shows the probability that k nodes confirm a registration when the replication factor is $r = 5$. The dotted line shows $h(5, p_n) = g(5, 5, p_n) + g(5, 4, p_n) + g(5, 3, p_n)$.

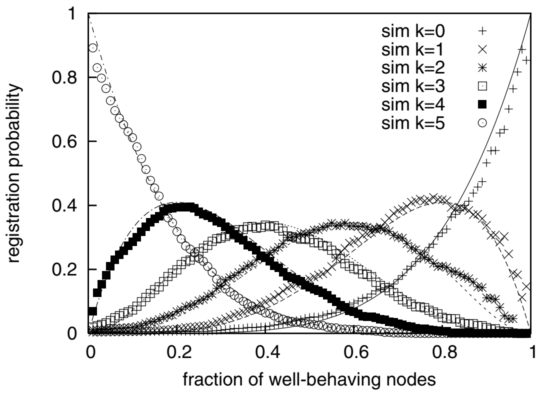

Figure 4. Simulation results

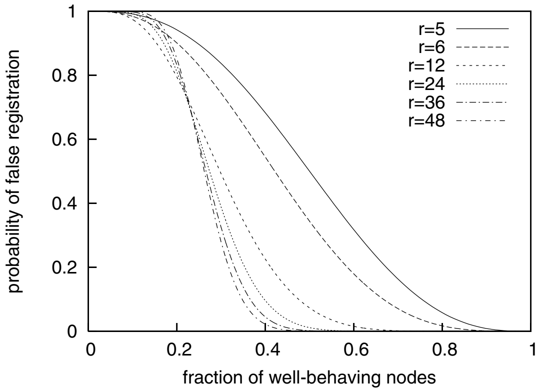

Figure 5. Ratio between 'false registering' and number of malicious nodes for different replication factors

This sum is the probability that a malicious participant can register more than the allowed number of nodes.

Figure 4 shows the corresponding simulative results for $n = 250$ nodes and $p = 0.01$ until $p = 0.99$ in 1% steps. For comparison reasons we also depicted the analytic graphs, which are the same as in Figure 3. In contrast to the first presented simulation, malicious participants in this case enter the network with a nodes and just try afterwards if it would be possible to join, but they do not join the network. If the nodes would join the network, they would perform a Sybil attack and therefore make it impossible to calculate the probabilities.

When we increase the replication factor r the probability that a malicious node can successfully 'false register' decreases. Figure 5 shows graphs for different r. These graphs correspond to the function $h(r, p_n)$.

Figure 6 depicts the fraction of well-behaving nodes necessary depending on the replication factor, if we allow a false registration probability of $h(r, p_n) \leq 0.001$. The results are calculated numerically. The small ups and down

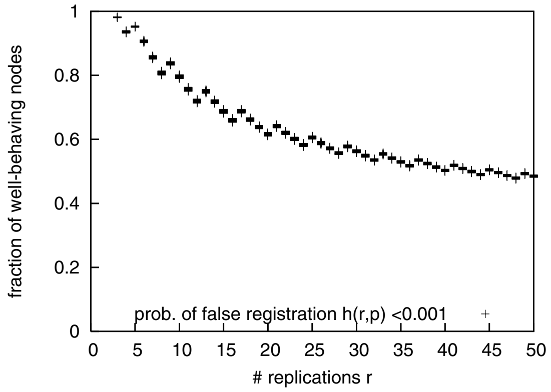

Figure 6. Maximal ratio of malicious nodes vs. number of replications

result from the ceiling function $\lceil \frac{r}{2} \rceil$. It is obvious that the graph tend towards a probability of $p_n = 0.5$. This is because at least 50% of the nodes have to confirm the correct registration.

The results indicate that the approach might provide a level of Sybil-resistance acceptable for some P2P networks when the parameters r and a are adjusted appropriately. The probability of 'false registrations' can be arbitrarily lowered towards zero by increasing the number of replications. Clearly, increasing r also results in an increased communication overhead.

7 Conclusion and outlook

We presented a taxonomy for the identifier assignment and verification process in P2P networks and pointed out the challenges of time-stability and identity differentiation by clearly distinguishing between network nodes and participants. The fundamental results by Douceur indicates that a truly Sybil-proof P2P network needs a central entity for *ID* assignment. We addressed the question left open in Douceur's work for the level of Sybil-resistance that could be achieved by a decentralized *ID* assignment procedure.

We have proposed a registration procedure called self-registration that is designed to be a natural extension of the P2P mechanism. We presented a first set of results that show that the proposed procedure might be able to effectively regulate the number of nodes per participant.

Many open questions exist in the area of 'Sybil resistant' distributed mechanisms. First of all, the questions of how long the time period lasts in which one can safely assume the network to be dominance-free has to be addressed. Second, we also have to analyze improved distribution and verification mechanisms for the self-registration approach to further improve its Sybil resistance level.

References

[1] K. Aberer, A. Datta, and M. Hauswirth. Efficient, self-contained handling of identity in Peer-to-Peer systems. *IEEE Trans. on Knowledge and Data Engineering*, 16(7), 2004.

[2] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proc. of the 5th USENIX Symposium on Operating Systems Design and Impl.*, Boston, MA, USA, 2002.

[3] A. Cheng and E. Friedman. Sybilproof reputation mechanisms. In *Proc. of the 2005 ACM SIGCOMM workshop on Economics of peer-to-peer systems*, pages 128–132, 2005.

[4] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson. Sybil-Resistant DHT Routing. In *Proc. of Computer Security (ESORICS '05): 10th European Symp. on Research in Computer Sec.* Springer, 2005.

[5] J. Douceur. The Sybil Attack. In *1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*. Springer,, 2002.

[6] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 228–236, New York, NY, USA, 2004. ACM Press.

[7] A. Fiat, J. Saia, and M. Young. Making Chord Robust to Byzantine Attacks. In *Proceedings of Algorithms - ESA 2005: 13th Annual European Symposium (LNCS 3669)*, pages 803–814. Springer, 2005.

[8] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[9] R. C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, 1978.

[10] T. Narten and R. Draves. Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 3041, Jan. 2001.

[11] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In *Proc. of the third int. symposium on Information processing in sensor networks*, pages 259–268, New York, NY, USA, 2004.

[12] A. Rowstron and P. Druschel. Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems. In *Proc. of Int. Conf. on Distributed Systems Platforms (Middleware)*, pages 329–350, Heidelberg, Germany, 2001.

[13] E. Sit and R. Morris. Security Considerations for Peer-to-Peer Distributed Hash Tables. In *Peer-to-Peer Systems: 1st Int. Workshop (IPTPS '02)*, pages 261–269, 2002.

[14] M. Srivatsa and L. Liu. Vulnerabilities and security threats in structured overlay networks: a quantitative analysis. In *Proceedings of Computer Security Applications Conference, 2004. 20th Annual*, pages 252–261. Springer, 2004.

[15] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, pages 149–160. ACM Press, 2001.

[16] H. Tyan. J-Sim. http://www.j-sim.org/, DRCL, The Ohio State University, 2005.