Identity Crisis: Anonymity vs. Reputation in P2P Systems*

Sergio Marti and Hector Garcia-Molina
Stanford University
Stanford, CA 94305
{smarti, hector}@cs.stanford.edu

Abstract

The effectiveness of reputation systems for peer-to-peer resource-sharing networks is largely dependent on the reliability of the identities used by peers in the network. Much debate has centered around how closely one's pseudo-identity in the network should be tied to their real-world identity, and how that identity is protected from malicious spoofing. In this paper we investigate the cost in efficiency of two solutions to the identity problem for peer-to-peer reputation systems. Our results show that, using some simple mechanisms, reputation systems can provide a factor of 4 to 20 improvement in performance over no reputation system, depending on the identity model used.

1. Introduction

The increase in wide-area network bandwidth and commodity computing power has promoted the use of peer-to-peer (P2P) systems. P2P systems allow any user to share resources while maintaining autonomy and independence from centralized servers, thus improving availability and fault tolerance. However, while it may be possible to trust a single centralized service, it is unwise to trust the multitude of anonymous service providers which populate P2P systems. Verifying the validity of the resources offered by other peers is often expensive in terms of time as well as human and computer resources.

As users access resources from other peers, they develop opinions about the trustworthiness of those peers. These opinions are collected and shared through certain mechanisms, called reputation systems. Specifically, reputation systems collect, process, and propagate information about the reliability of resource providers in the network. This information serves as a reference to identify which peers are likely to provide acceptable service.

Maintaining statistics of node behavior requires some form of persistent node identification. In order to build reputation, a user or node must have some form of identity which is valid over a period of time. The longer this period of time, and the more resistant the identity is to spoofing, the more accurately the reputation system can rate nodes [13].

The simplest way to identify a node is to use its IP address. This method is severely limited because addresses are vulnerable to IP-spoofing and peers are often dynamically assigned temporary IP addresses by their ISPs. Instead, a more reliable method may be to use self-signed certificates. This technique allows well-behaved nodes to build trust between each other over a series of disconnections and reconnections from different IP addresses. Although malicious nodes can always generate new certificates making it difficult to distinguish them from new users, this technique prevents them from impersonating existing well-behaved nodes.

Some argue that the only effective solution to the identity problem in the presence of malicious nodes is to use a central trusted login server, which assigns a node identity based on a verifiable real-world identity [1]. This would limit a malicious node's ability to masquerade as several nodes and to change identities when their misbehavior is detected. It would also allow the system to impose more severe penalties for abuse of the system.[1]

In this paper, we study the different approaches reputation systems for peer-to-peer file retrieval must take either with a central login or with self-managed identities. We analyze the performance of each scenario and compare it to the base case with no reputation system. We look at how new nodes should be treated, and present some mechanisms to further improve system efficiency. The results presented in this short paper are a subset of the system and threat models we have studied. Here we do not discuss all options and issues; more information is available in the extended version of this paper [12].

In Section 2 we present our system model and its assumptions. Then, Section 3 discusses the reputation sys-

*This research is supported in part by NSF Grant (IIS-9817799).

[1] For example, a person might have to use a valid credit card to enter the system, allowing the system auditors to debit their card if they misbehave.

Proceedings of the Third International Conference on Peer-to-Peer Computing (P2P'03)
0-7695-2023-5/03 $17.00 © 2003 IEEE

tems used in the experiments and their options. Section 4 describes the metrics used for evaluating and comparing the reputation systems. In Section 5 we specify the details of the simulation environment used for the experiments, the results of which are discussed in Section 6. Section 7 provides an overview of related work. Finally, we conclude in Section 8.

2. System Model

A peer-to-peer file retrieval network is composed of n peer nodes, N_1 ... N_n, arranged in an overlay network. In a file retrieval system each node, N_i, stores a collection of files, F_i. When a node wishes to retrieve a file from the system it queries the system, collects responses, and selects a copy to present to the user. The query is propagated through the network up to a certain number of hops, specified by a Time-To-Live (TTL). Any node that receives the query will check if it has a copy of the file. If so, it replies to the query originator, who then selects a responder from which to fetch the file.

Unfortunately, many peers in the network may send back incorrect files either maliciously or accidentally. For example, people who disagree with Darwin's theory of evolution may provide copies of his book, *Origin of Species*, with key passages missing or edited. Other peers may download this version and, unaware of modifications, share it, further propagating the inauthentic (or fake) version.

What does it mean for a file to be invalid or fake? The issue of file authenticity is discussed in the following section. The behavior of peers in the system with respect to the authenticity of the files they send each other is captured in the threat model, which is discussed in Section 2.2. Reputation systems, which track node behavior in order to mitigate the problem of inauthentic files are covered in Section 3.

2.1. Authenticity

In our model the unit of storage and retrieval is the document. Every document D consists of some content data C_D and metadata M_D which uniquely describes the content. If two documents contained the same metadata but different content, there must be some information pertaining to their differences that should be included in the documents' metadata to make them unique. For example, different editions of a book should include the edition and the year published in their metadata. Below we illustrate a sample document for a specific edition of the Dickens' novel *A Tale of Two Cities*. If the only metadata provided were the title and author, the document may not be unique, since the other editions of the same book exist in other languages, or may include notes or pictures.

Metadata
Title: A Tale of Two Cities
Author: Charles Dickens
Publish Date: April 2002
Publisher: Barnes & Noble Books
...
Content
It was the best of times, it was the worst of times...

In general, a document is considered authentic if and only if its metadata fields are "consistent" with each other and the content. If any information in the metadata does not "agree" with the content or the rest of the metadata, then the document is considered to be inauthentic, or fake. In the example above, if the Author field were changed to Charles Darwin, this document would be considered inauthentic, since Barnes & Noble Books has never published a book titled *A Tale of Two Cities* written by Charles Darwin that begins "It was the best of times..." But determining consistency of a document is largely dependent on the application of the document by its user.

Since there is no one domain-independent definition of authenticity, we simply assume the existence of a global authenticity function, $A(D)$ which enables one to verify the authenticity of a document D. Evaluating the authenticity function is likely to be very expensive and may require human user interaction or even a third party. An example of human verification would be if Alice were to download a song from a music sharing service, she could determine whether it is the correct song by listening to it. Therefore, it is crucial to reduce the number of times $A(D)$ is evaluated. The goal of a reputation system is to select the best source for a document so as to reduce the likelihood of fetching an inauthentic copy and wasting the user's time evaluating the authenticity function on a bad file.

2.2. Threat Model

Malicious nodes may reply to any request with a fake (inauthentic) file. They may reply with false data to all queries, to certain percentage of random queries, or only when a specific document is requested. Bad nodes may also lie when sharing information with other nodes and may collaborate with other malicious nodes to promote the propagation of their fake files. We also assume that well-behaved nodes always verify the authenticity of any document they have before sharing it with other peers. This assumption may be unrealistic for many peer-to-peer systems. We have performed experiments in which a small fraction of the documents provided by good nodes were invalid. This modification had little effect on the results presented here.

For simplicity, our model generally assumes that all nodes use the same identity for their lifetime. This mim-

ics a system with a centralized login server, assigning unforgeable IDs based on real-world identities. To emulate an environment of self-managed identities we study malicious nodes altering their identities in order to hide their behavior from the network. This is modelled by erasing all information gathered on a malicious node after it sends an invalid document to the query source node for verification. If node M sends node S a fake document, all information collected by nodes (including S) about M is erased. Essentially all nodes "forget" about bad nodes. We abbreviate the references to the login server and self-managed identities scenarios as *Login* and *Self-Mgd*, respectively.

Our threat model for this paper assumes that specific documents in the system have been targeted by all malicious nodes. We call the set of documents that bad nodes are intentionally disseminating forgeries the *subversion set*. The percentage of nodes in the network that are malicious is specified by the parameter π_B. We assume there is no correlation between a document's popularity in the system and its likelihood to be a target for subversion. Therefore, each unique document has an equal probability of being in the subversion set, given by the parameter p_B. Bad nodes also have documents, not in the subversion set, of which they provide authentic copies. In Section 6.3 we study the performance effects of varying both π_B and p_B.[2]

3. Reputation Systems

As nodes interact with each other, they record their experiences and develop an opinion, or *reputation rating*, for each other. For example, a node may keep track of the total number of documents it received from each node, and how many were authentic. The collection of reputation-related information a node has gathered can be modelled as a *reputation vector* of length n, where n is the total number of nodes in the network. Initially, all entries are *undefined* since no node has interacted with any other node. As nodes exchange documents they record statistics about each other based on the quality of the documents received. Nodes may also share their opinions about other nodes with each other. A node uses the statistics it collects and the opinions of other nodes to maintain reputation ratings for its peers. The reputation vectors can be arranged into an $n \times n$ *reputation matrix*, R, where the ith row is node i's reputation vector. Cell $R_{i,j}$ would contain node i's "opinion" of node j. Since there is likely to be little interaction between nodes compared to the total size of the network, R will be sparse.

When a node retrieves a document, the search follows three steps. First, the node queries the system for the document it desires. Then, it collects all replies (and their source IDs) in a *response set*. Finally, the node repeatedly selects responses from the set to fetch and verify (using the authenticity function) until an authentic copy of the desired document is found. In the selection stage of the search a reputation system performs three basic operations. First, a *decision procedure* takes the query response set and the node's reputation vector and chooses a document copy from the responses. This is usually the copy the procedure determines is most likely authentic, but not necessarily.[3] Second, the authenticity function is calculated on the selected document copy. This may be done programmatically if possible, but most likely requires presenting the document to the user who then accepts or rejects the document after analyzing it. Third, based on the result of the authenticity function, the reputation system updates the reputation rating for the peer whose response was chosen. If the authenticity function rejected the document then the reputation system repeats the three steps until a copy is accepted, there are no more responses, or the decision procedure decides there are no more responses worth selecting (for example, if the reputations of the nodes that sent the remaining replies are too low).

In this paper we consider several variants of a simple reputation system where nodes do not share their opinions with other peers. We compare them to a file retrieval system that uses no reputation information at all.

Random Selection: The random algorithm is the base case, randomly choosing from the query responses until an authentic document is located. No knowledge or state about previous interactions is kept or used.

Local Reputation System: The first reputation system we consider maintains information about prior interactions and uses the data when selecting a source for future query results. No information is shared with other peers. The information kept for each node is the total number of times the authenticity of a document from that node was checked, and the number of times it proved to be authentic. Reputation ratings are calculated by dividing the number of verified authentic responses by the total number of responses checked. This results in a rating ranging from 0 to 1, with 0 meaning no authenticity check passed and 1 meaning all authenticity checks passed. Two selection procedures are evaluated for the system:

- The *Select-Best* decision procedure selects the response from the node(s) with the highest trust value. If the selected response is inauthentic, the procedure is called again and the next highest rated node's response is chosen.
- Select-Best may overload the more reputable peers, so an alternative is to spread out file requests. The *Weighted* decision procedure probabilistically selects

[2] Other threat models we have studied are discussed in [12].

[3] The heuristic used may select probabilistically, or may encourage testing unknown nodes.

Table 1. Metrics and simulation statistics

Metric	*Description*
q_{tot}	# of queries generated
q_{good}	# of queries with an authentic document in at least one response
q_{succ}	# of successful queries where the selection procedure located an authentic document
A_i	# of authenticity function evaluations performed on documents from node i
A	# of authenticity function evaluations
r_v	Verification ratio
r_{miss}	Miss rate

the document to analyze weighted by the document provider's trust value. If nodes N_i and N_j both provide replies to node N_q and $R(q,i) = 0.1$ and $R(q,j) = 0.9$, then N_j is nine times as likely to be chosen as N_i.

Since all entries in R are initially undefined, an initial reputation rating ρ_0 must be assigned to nodes for which no statistics are available, to be used for comparing response nodes in the selection stage. Analysis of different values for ρ_0 is provided in Section 6.1.

In some domains it may be easy for malicious nodes to automatically generate fake responses to queries. In situations where a node is querying for a rare document, it may receive many replies, all of which are bad. To prevent the node from fetching every false document and calculating $A(D)$, we introduce a *selection threshold value* (ρ_T). Any response from a node whose reputation rating is below this threshold is automatically discarded and never considered for selection.[4] In Section 6.2 we analyze the effects of varying the threshold value on performance.

4. Metrics

When studying reputation systems it is necessary to determine what metrics best measure the success of a particular system. In this section we describe the statistics we gather for each experiment and the metrics we compute using the statistics. They are summarized in Table 1.

From among all the queries generated during execution (q_{tot}) we are specifically interested in the number of good queries (q_{good}) and the number of successful queries (q_{succ}). A *good query* is any query whose response set includes at least one authentic copy of the queried document, even if no authentic copy was located by the selection procedure. A *successful query* is a query that results in an authentic copy of the requested document being selected by the selection procedure. The relation between the three statistics is given by the following equation:

$$q_{tot} \geq q_{good} \geq q_{succ} \tag{1}$$

For the reputation systems we are testing, if q_{succ} always equals q_{good} then the system is considered to be 100% effective.

4.1. Efficiency

The main objective of a reputation system is to reduce the number of documents the user must examine before finding the correct document for their query. We call this the *efficiency* of the reputation system. This is equivalent to minimizing the number of times the authenticity function is calculated in the selection stage. This metric seems the most practical and direct measure of a particular selection heuristic's performance.

To compare reputation system efficiency we are interested in the total number of authenticity function evaluations incurred by the system, A. During execution we record the number of authenticity checks on documents supplied by each node N_i, which we refer to as A_i. From these statistics we compute the total number of authenticity function evaluations as

$$A = \sum_{i=1}^{n} A_i \tag{2}$$

Three comparison metrics are used in the results section based on the metrics described in Section 4. The first is the number of total authenticity function evaluations (A) divided by the number of successful queries (q_{succ}). We call this the *verification ratio* (r_v).

$$r_v = \frac{A}{q_{succ}} \tag{3}$$

The best possible performance would be a prescient selection algorithm which would choose the authentic copy first and present it to the user for verification. If no response contains an authentic copy, then none of the responses would be chosen for verification. This would give a metric value of 1.

4.2. Effectiveness

While systems reduce the number of document fetches and authenticity function computations to be more efficient, it often comes at the sacrifice of *effectiveness*. The effectiveness of a search system relates to its ability to locate an answer, given that one exists somewhere in the network. A reputation system's effectiveness can be considered to be the fraction of queries for which an authentic document is selected, given that one exists in the response set. We

[4] New nodes are automatically exempt from being discarded, even if $\rho_0 < \rho_T$.

call this metric the *miss rate*. This measurement of effectiveness is only accurate for systems in which the reputation algorithm does not interfere with query response or query/response propagation, but it is accurate for the systems described here.

Some reputation systems with selection thresholds may not locate an authentic document even when one is available, and thus are not completely effective. We are interested in measuring how often such systems report a failure to a good query. We introduce the *miss rate* (r_{miss}), given by the equation

$$r_{miss} = \frac{q_{good} - q_{succ}}{q_{good}} \quad (4)$$

The miss rate gives the fraction of good queries that were missed. A system which returns a valid document for every good query will have a miss rate of 0. A system which never returns a good response would have a miss rate of 1. Therefore, the miss rate is inversely related to the effectiveness of the reputation system.

5. Simulation Details

To compare the reputation systems described above under varying threats and conditions we prepared a simulator based on our system description. The simulations were run on a Dual 2.4Ghz Xeon processor machine with 2GB of RAM. All simulations were conducted on our own extensible P2P Simulator. Each data point presented in the results section represents the average of 12 simulation runs with different seeds.

Studies of peer-to-peer systems such as Gnutella [6] have shown that peer-to-peer network topologies follow a power-law distribution. [4] We use a randomly generated fully connected power-law network with n nodes, a maximum node degree of d_{max} and an average node degree of d_{avg}. Unless stated otherwise $n = 10000$, $d_{max} = 150$ and $d_{avg} \approx 3.1$. For simplicity we assume the network stays static for the duration of the simulation and nodes do not enter, leave, or move within the network (though, as stated above, malicious may change their identities).

For the experiments in which the reputation systems do not share information between peers or affect each other's queries in any way, a random node is chosen at the start and a query is sent from that node in each timestep and evaluated completely before the next query. Therefore a simulation run of 1000 (default length), refers to the number of queries sent and processed by the single source node. When simulating reputation systems where peers exchange reputation information, a random node is chosen at each timestep as the query source for 100,000 timesteps.

Each node was assigned a number of unique shared files, chosen at random from the distribution of shared files collected by Saroiu et al [14]. Each node is then assigned its unique documents based on a Zipf distribution. Our query model is similar to that proposed in [16]. One modification is to use Zipf distributions instead of exponential distributions for both the query popularity and query "selection power" distributions based on findings in recent research [15], though the results are equivalent for our evaluations. Both Zipf distributions have a maximum rank of 100000, corresponding to 100000 unique documents in the system. The query popularity distribution uses an α of 0.63 from rank 1 to 250 and an α of 1.2 above 250. This two-part Zipf distribution better models observed query popularity in existing peer-to-peer networks [15]. The file distribution follows a Zipf distribution with $\alpha = 1.2$.

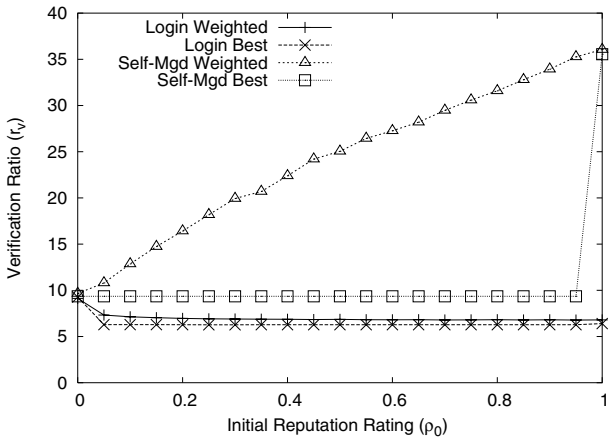

Figure 1. Efficiency for varying ρ_0. Lower value is better. 1 is optimal.

6. Results

In this section we address several of the questions brought up in the previous sections. Specifically:

1. Is an initial reputation rating of zero always preferable to nonzero?
2. What is the cost in efficiency (as defined here) for using self-managed identities in lieu of a trusted login server?
3. Is there a benefit to using a selection threshold?

Primarily we focus on the results for the local reputation system experiments and briefly summarize the results for a voting-based reputation system.

6.1. New Node Reputation

In this experiment we varied the initial reputation rating (ρ_0) used by the local reputation system for any node from which we have not received a document and checked its authenticity. Our experiments demonstrate that, though a

reputation system performs similarly for both identity models for a ρ_0 of 0, efficiency in the login server scenario can improve substantially by increasing ρ_0, while performance in the self-managed identities scenario will only worsen.

Figure 1 shows that for the *Login* scenario, a nonzero initial reputation rating (eg. $\rho_0 = 0.4$) performs better by a factor of 1.5 in terms of minimizing the number of authenticity checks computed. If malicious nodes cannot change their identities to pose as new nodes after misbehaving, there is a benefit to selecting new nodes over previously encountered malicious nodes.

If malicious nodes are allowed to change their identities, as in the self-managed identities scenario, they will usually be treated as new nodes with a reputation rating of ρ_0 in the selection procedures. We would expect that varying ρ_0 would have a significant effect for *Self-Mgd*. Figure 1 shows that increasing ρ_0 decreases the efficiency when using the Weighted procedure, though unexpectedly, the Select-Best procedure is not affected (until $\rho_0 = 1$). For example, from a ρ_0 of 0.0 to 0.5, the verification ratio (the average number of authenticity checks performed per query) of the Weighted method goes from 9.7 to 25.1, while Select-Best stays constant at 9.4. Since the Weighted method considers all nodes (weighted by their ratings) in the selection stage, it is important to lower the weight of new nodes, which are more likely to be malicious nodes in the scenario of self-managed identities than in that of a login server. The results support our intuition. The Select-Best method's unvaried performance across all values of ρ_0 can be attributed to the fact that often a node receives a reply from a peer which has previously provided an authentic document, in which case the node will always choose the reputable source over any unknown peer.

From these experiments we selected 0.3 as the default value for ρ_0 for *Login*. Many of the following experiments were additionally performed with other values of ρ_0, but the results did not vary noticeably from those at $\rho_0 = 0.3$ and are not discussed. For *Self-Mgd* simulations we use only ρ_0 = 0, which clearly performed best for the Weighted method.

6.2. Selection Threshold

Figure 2 shows tests varying the value of the selection threshold for both the Weighted and Select-Best variants of the local reputation system. The verification ratio is plotted as a function of ρ_T. As stated above, ρ_0 was set to 0.3 for *Login* and 0 for *Self-Mgd*.

The result is surprising. For *Login* all values of ρ_T above 0 resulted in almost equal performance, yet significantly better than ρ_T = 0 (r_v of 6.3 for ρ_T = 0 down to 2.0 for $\rho_T > 0$).[5] Because malicious nodes always reply with a copy when a document in the subversion set is queried for, the vast majority of responses in the response set come from malicious nodes supplying bad copies. When searching for rare content, it is common to receive only bad copies from malicious nodes. The threshold prevents nodes from repeatedly fetching and testing documents from peers which have proven malicious or unreliable in the past. The drawback of the selection threshold is a decrease in query effectiveness (discussed in the following section).

[5] Though almost the same, the values of r_v for different nonzero ρ_T for a given reputation system variant are not exactly identical.

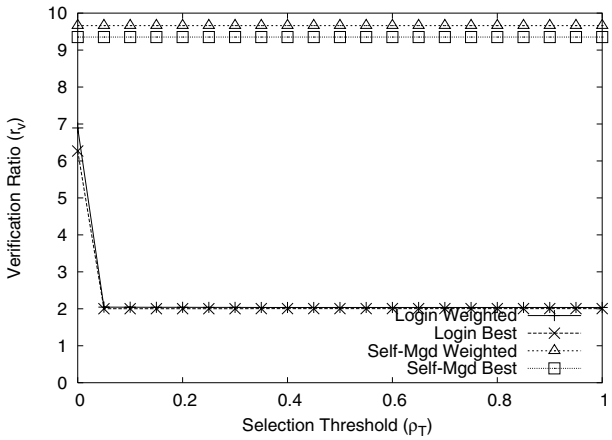

Figure 2. Varying selection threshold values.

For *Self-Mgd* varying ρ_T had no effect. Remembering which nodes have lied in the past is of no use if those nodes can immediately change their identities to hide their misbehavior. The threshold may be useful if nodes were motivated to maintain their identities, perhaps by providing incentives for building reputations.

In successive tests any system variant using a selection threshold uses a ρ_T value of 0.2 unless otherwise stated.

6.3. Performance under Various Threat Conditions

In this section we look at system performance under different threat model parameter values. Specifically we demonstrate how overall efficiency is affected by varying the percentage of malicious nodes in the system (π_B) and the probability of a unique document being in the subversion set (p_B). Eight different variants of the local reputation system were tested. These eight variants are derived from three system parameters: the identity model (*Login* or *Self-Mgd*), ρ_T (0 or 0.2), and the decision procedure (Weighted or Select-Best).

The graphs in Figure 3 present the system performance for varying π_B and p_B. The results show that overall a trusted login server significantly reduces the cost of insuring authenticity over self-managed identities roughly by a factor of 5.5. Yet, using a reputation system with the *Self-Mgd* model outperforms having no reputation system at all

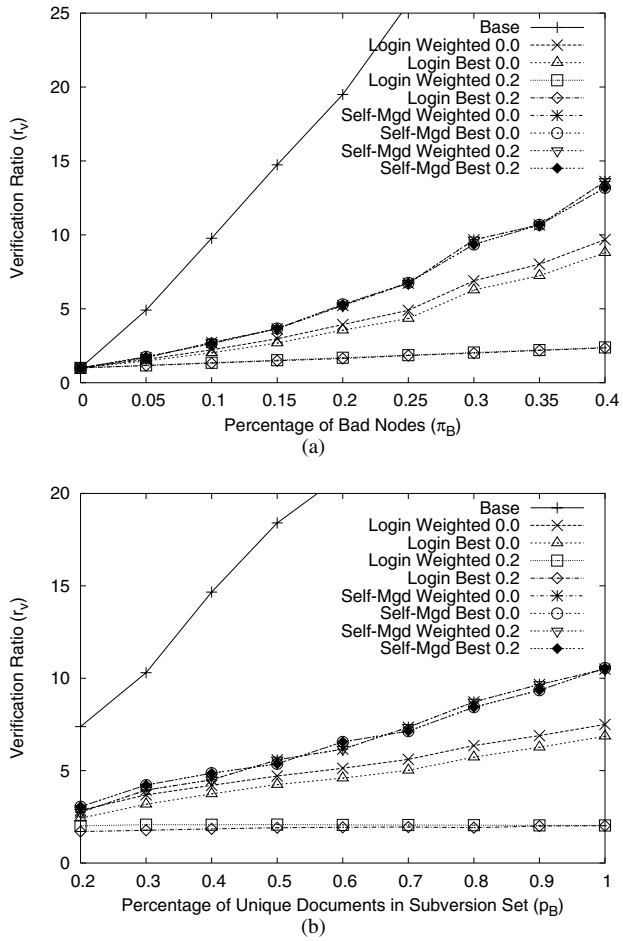

Figure 3. Efficiency comparison.

(Base curve in Figure 3) by an additional factor of 3.5.

For both graphs, the curve corresponding to the base case, of purely random selection, quickly climbs out of the range of the graphs. In Figure 3(a) the base curve increased steadily to 46 at π_B = 0.4; 3.5 times the verification ratio of the *Self-Mgd* variants, and up to 20 times the r_v of *Login* using a selection threshold. In Figure 3(b) the base curve climbed to 35 at p_B = 1; resulting in 3.4 times the r_v of the *Self-Mgd* variants, and approximately 17 times that of *Login* with ρ_T = 0.2. This means one would expect to have to fetch and test on average 20 times as many query responses in order to find a valid response! Even using self-managed identities, a rudimentary reputation system provides significant performance improvements over no reputation system. Even then users would expect to fetch over ten bad copies for every good copy they locate (for $\pi_B > 0.3$). In contrast, a peer using a selection threshold in a login server environment would only expect to encounter one or two fakes for every authentic file, no matter the level of malicious activity in the network.

Figures 3(a) and 3(b) show that the Select-Best and the Weighted procedures perform similarly. Overall the Select-Best method outperformed the Weighted method, especially in the *Login* model. Though the Select-Best performed well and served to mitigate the performance variance of other parameters (such as the initial reputation rating), it does have drawbacks. A study of the load on well-behaved nodes (measured as the number of documents fetched *from* a node) showed a much more skewed distribution for the Select-Best variants than the Weighted variants. In fact, the highest loaded good nodes in the Select-Best simulations were being asked for 2.5 times as many documents as the highest loaded nodes in the Weighted simulations.[6] At the bottom of the distribution, hundreds of nodes were never accessed in the Select-Best simulations that were in the Weighted simulations. This dramatic skew in load distribution can result in unfair overloading, especially in a relatively homogeneous peer-to-peer network.

Both graphs illustrate that the selection threshold is useless in the *Self-Mgd* scenarios, but provides a large performance boost for *Login*. This supports our findings in the previous section, and demonstrates it was not an artifact of the selected values of the threat parameters. Using a selection threshold system efficiency is relatively unaffected by variations in π_B and p_B.

Measurements of effectiveness in these experiments (only applicable to a nonzero selection threshold) resulted in a miss rate well below 0.001 (0.1 of 1%) for the experiments varying π_B at a constant p_B of 0.9. For the experiments varying p_B, the miss rate increases as p_B decreases, but always remains below 0.0025 (0.25 of 1%). As p_B decreases, the subversion set decreases. Because malicious nodes become more likely to provide authentic documents, but tend to fall under the threshold, the effectiveness of the system deceases. For most applications these miss rates are acceptable, especially when compared to the increased efficiency offered by the selection threshold.

6.4. Voting System

In addition to the system described above that uses only locally collected statistics, we have simulated a voting-based reputation system. In this system, the querying node asks a number of peers for their opinions of the query responders. Combining each peer's opinion with local statistics (if any exist), the node rates the responders and applies the same decision procedure variants as we have described above.

Conducting the same experiments using the voting-based reputation system resulted in similar relative performance differences, though with less variation than seen above. Other factors exclusive to the voting scheme, such as the

[6] Due to size constraints, a more extensive discussion of load-balancing must be omitted. Please refer to [12] for more information.

weight placed on local statistics versus peer opinion, caused greater fluctuations in system efficiency. A detailed evaluation of the voting reputation system and its parameters is presented in [12].

7. Related Work

There has been extensive research on reputation in general (eg. [7] [8] [11]). In the area of peer-to-peer systems recent work has focussed on proposing interesting reputation systems (eg. [2] [9] [10]). Most assume one type of identity infrastructure and is specifically geared towards that model. Little research has gone into comparing different practical identity models. Much of this work has been purely analytical in a more general context than the simulations we have presented. We briefly mention a few such papers.

Reference [5] presents a game theoretical model, based on the prisoner's dilemma, for analyzing the social cost of allowing nodes to freely change identities. It proposes a mechanism, based on a centralized trusted intermediary. It ensures each user is assigned only one system identifier, yet protects their anonymity so that even the intermediary does not know which identifier was assigned to which node.

In [3] Douceur discusses the problem of preventing users from using multiple identities in a system with no trusted central agency (the Sybil attack). He presents methods for imposing computational cost on identity creation and lists system conditions necessary to limit the number of identities peers can generate.

8. Conclusion

We have compared two practical identity infrastructures for peer-to-peer resource-sharing environments. A centralized trusted login server that ties nodes' network pseudo-identities to their real-world identities provides better support for reputation systems by preventing nodes from quickly changing identities. However, this benefit comes at a high management cost and requires users to disclose information to a level which they may not find acceptable. The decentralized approach, where each node generates its own identity, provides a higher level of anonymity while simultaneously preventing identity hijacking, at the cost of no enforced identity persistence for malicious nodes. Though we have concentrated on two distinct identity models, many practical solutions fall in a spectrum between them (such as providing incentives for persistent identities) and perform accordingly.

Our results show that even simple reputation systems can work well in either of the two identity schemes when compared to no reputation system. In environments where system identities are generated by the peers themselves, all unknown nodes should be regarded as malicious. But, if a centralized login authority enforces identities tied to real world identities, then the optimal reputation for unknown nodes is nonzero. In addition, certain techniques, such as using a selection threshold, provide large benefits in efficiency for one identity scheme, but are ineffectual for others.

References

[1] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Security for structured peer-to-peer overlay networks. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, 2002.

[2] E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 207–216. ACM Press, 2002.

[3] J. R. Douceur. The Sybil Attack. In *Proceedings of the International Workshop on Peer-to-Peer Systems*, 2002.

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

[5] E. Friedman and P. Resnick. The social cost of cheap pseudonyms. *Journal of Economics and Management Strategy 10*, (2):173–199, 1998.

[6] Gnutella specification. www9.limewire.com/developer/gnutella_protocol_0.4.pdf.

[7] B. A. Huberman and F. Wu. The dynamics of reputations. www.hpl.hp.com/shl/papers/reputations/, 2002.

[8] R. Jurca and B. Faltings. Towards incentive-compatible reputation management. In *Proceedings of the AAMAS 2002 Workshop on Deception, Fraud and Trust in Agent Societies*.

[9] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the Twelfth International World Wide Web Conference*, 2003.

[10] K. Lai, M. Feldman, I. Stoica, and J. Chuang. Incentives for cooperation in peer-to-peer networks. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[11] R. Marimon, J. Nicolini, and P. Teles. Competition and reputation. In *Proceedings of the World Conference Econometric Society*, 2000.

[12] S. Marti and H. Garcia-Molina. Examining metrics for reputation systems (in progress). Technical report, 2003. dbpubs.stanford.edu/pub/2003-39.

[13] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. Communications of the ACM, pages 45-48, December 2000.

[14] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN '02)*, San Jose, CA, USA, January 2002.

[15] K. Sripanidkulchai. The popularity of gnutella queries and its implications on scalability. Featured on O'Reilly's www.openp2p.com website, February 2001.

[16] B. Yang and H. Garcia-Molina. Comparing hybrid peer-to-peer systems (extended). Technical report, 2000.