Economic Measures to Resist Attacks on a Peer-to-Peer Network

David S. H. Rosenthal, Mema Roussopoulos, Petros Maniatis, Mary Baker
Stanford University, Stanford, CA 94305-9040
http://lockss.stanford.edu/

***Abstract*— Peer-to-peer systems in which the peers are truly autonomous have valuable properties, including resistance to certain forms of organizational failure and legal attack. Unfortunately, they can be vulnerable to malign peers.**

In the context of the LOCKSS system, a peer-to-peer digital preservation system for e-journals, we describe a set of techniques that enable a large population of autonomous peers to resist attack by a substantial minority of malign peers endowed with unlimited computational resources. LOCKSS peers are able to detect attacks and alert the community of peer operators before damage becomes irreversible. These techniques include rate limitation and making peers "pay" certain costs by demanding proofs of effort from them.

1. Introduction

We are investigating ways of building systems from large numbers of autonomous, unreliable, mutually suspicious peers that can sustain high probabilities of meeting specified service levels over long periods of time. During these times, such systems must expect to be attacked and subverted. Our initial test-bed for these ideas is the LOCKSS™ (Lots Of Copies Keep Stuff Safe) system, a peer-to-peer digital preservation system for academic journals published on the Web. An initial version [18] has been under test for several years at about 50 libraries world-wide.

Scientific communication has transitioned to the Web. In particular, much peer-reviewed science now appears only in e-journal form [12]. Academic journals are funded by university and other librarians paying institutional subscription rates. The librarians consider it part of their job to preserve access to the record of science for future generations. The transition to the Web has meant a transition from a purchase model, in which librarians buy and own a copy of the journal, to a rental model, in which they rent access to the publisher's copy. Rental provides no guarantee of future access, and librarians fear the worst [15].

The LOCKSS program is implementing the purchase model for the Web, providing tools librarians can use to take custody of, and preserve access to, web-published materials. These tools allow libraries to run persistent web caches (built from low-cost, unreliable, off the shelf hardware and free, Open Source software) that:

- *collect* material by crawling the e-journal Web sites,
- *distribute* material as a proxy cache does, to make it seem to a library's readers that the material is still available at its original URLs, even if it is no longer available there from the original publisher [19], and
- *preserve* material in cooperation with other caches in a peer-to-peer network by, at intervals, having samples of the peers vote in *opinion polls* on their content and thereby detect and repair damage.

We have recently designed and simulated an entirely new opinion polling protocol, which we plan to use as the basis for the production LOCKSS system. Despite significant design constraints, it shows an encouraging ability to resist attacks even by a substantial minority of malign peers endowed with unlimited computational power. It also detects these attacks and raises alarms to alert the community of peer operators before irrecoverable damage has been done.

In this paper we present a brief description of the design principles we use, the opinion polling protocol we developed using them, and the various strategies adversaries can use to attack it. On this basis we develop a research agenda to guide future work in this area. The full specification of the protocol can be found in [14].

2. Related Work

Superficially, the LOCKSS system may appear similar to peer-to-peer *storage* systems such as Free Haven [5], the Eternity Service [2], Oceanstore [13] and Intermemory [11]. In reality, the goals of the LOCKSS program are more limited:

- In P2P storage systems, peers cooperate to store data, with the peers as a whole accepting responsibility for storing enough copies to provide robustness. In the LOCKSS system each peer is responsible for obtaining and storing its *own* copy of each document it

wishes to preserve. Peers cooperate only to reduce the cost of preserving their copy, by detecting and repairing any damage caused by the poor reliability of low-cost hardware.
- Most P2P storage systems depend on long-term secrets, both as a basis for peer identity and as encryption keys to control access to the material they store. Long-term secrets are not appropriate for digital preservation [4]. They are highly likely to leak or be forgotten in the long time horizons for which preservation systems are intended.
- Some P2P storage systems strive for anonymity (e.g., Free Haven). The DMCA requires that LOCKSS caches have permission from the publisher to preserve their copyrighted content, making anonymity counter-productive.

Although it does not provide anonymity, the LOCKSS system has many similarities with peer-to-peer anonymity systems such as MorphMix [16], [17] and Tarzan [7]. In both cases, lack of central control is fundamental to achieving the system's goal of resisting a powerful adversary. These systems share the need for discovering peers, for using statistical techniques to detect the activities of malign peers, and for limiting the ability of a malicious adversary to degrade the system as a whole.

3. System Overview

Long-term digital preservation is an unusual application. It has very long time horizons, much longer than the mean time between failures of affordable off-the-shelf technologies. Libraries and their contents are subject to physical and organizational attack by powerful adversaries, including governments. It is not realistic to expect the system to survive all such attacks unaided, but it should have a high probability of detecting them before irrecoverable damage is done. The goal of the system is to prevent change to data, not to expedite it, so there is no need for speed. Therefore, the system must have a high probability of delivering correct content even under sustained attack.

The LOCKSS system consists of large numbers of unreliable, persistent web caches cooperating in a peer-to-peer network. The caches are installed in libraries in a wide range of countries and are *autonomous* and *equal*; there is no central control or administration to which they are subservient, nor are any peers more equal than others.

The design principles that have evolved from experience with this application differ in almost every respect from the conventional wisdom. We regard this not as a critique of the conventional wisdom as applied to more conventional applications, but as a fascinating opportunity to explore the utility of a different set of principles:

- *Limit the rate of operation*: nothing in the system should happen any faster than strictly necessary, which ensures that the system degrades as slowly as possible.
- *Assume a powerful adversary*: do not place arbitrary limits on the assumed adversary's capabilities. In particular, assume that, for example, by the use of flash worms [20], he can exert enormous efforts for limited periods from a potentially unlimited number of host addresses.
- *Do not keep secrets for long*: do not assume that peers are capable of keeping secrets for longer than a few days.
- *Do not depend on peer identity*: with no central control and no long-term secrets, peer identity can be spoofed at low cost. It must not be given significance, and should serve at most as a hint to help operations proceed.
- *Avoid third-party reputation*: relying on third-party testimony as to the bona fides of other peers renders a peer vulnerable to false or tampered testimony; this vulnerability is further aggravated by the lack of stable identities.
- *Curtail credit accumulation*: with neither stable identities nor a trusted "bank," histories of past behavior in the form of reputation or accumulated credit balances are not dependable. "Payment" for services must consist of proof of recent effort.
- *Minimize peer state*: all memories in the system are unreliable in our time frames, so the less state, and the shorter the time for which it has to be kept, the better.
- *Make intrusion detection inherent*: the system should exploit bimodal behavior [3] to trigger alarms when the adversary applies enough effort that there is a substantial risk of irrecoverable damage.

Peers using our opinion polling protocol keep a *reference list* containing a sample of the peer population. At intervals they choose a subset of the peers in the list and invite them to take part in a poll. The invitees exact a specified amount of otherwise useless computational effort from the caller of the poll using a *memory-bound function* (MBF) [1] scheme due to Dwork et al. [6]. On receipt of a proof of this effort they:

- nominate a sampling of peers from their reference list for possible inclusion in the caller's reference list,
- and construct a vote in a series of rounds each involving some otherwise useless MBF effort and a hash of the result of this effort with a part of the document.

When votes are complete they are sent to the caller, who verifies and tallies them to decide the result. If the votes indicate overwhelming disagreement with the caller's copy of the document, the caller repairs his copy. If the votes instead indicate overwhelming agreement, the caller assumes his copy does not require repair. Results between these two modes indicate coherent damage in the system and trigger an intrusion detection alarm. The caller also sends invitations to the nominated peers, computes the necessary MBF proofs, and verifies — but does not tally — their votes.

In each poll, voters are removed from the reference list; nominated peers who agreed with the result are then added to the list. The effort exerted by the nominated peers in constructing a vote is their price of admission to, and their "investment" in, the caller's reference list. This investment is important, because the protocol requires that a peer supply repairs only to peers in its list, to avoid theft. Entries in reference lists time out after a few polls.

By requiring proof of recent, directly-observed investment, we avoid two hurtful adversary behaviors: first, we prevent him from gaining a foothold in a reference list with only modest sustained effort over time, and second, we prevent him from affecting the system to a greater extent than his recent investment justifies. Both of these behaviors are possible when decisions are based on input measured on a scale that is independent of the size of peers' recent investment in the system [21].

Our assumed adversary is a conspiracy of malign peers. We are appropriately conservative for a preservation system, assuming for the purpose of simulation that the conspiracy starts when an implementation vulnerability common to a fraction of the peers allows the adversary to subvert that fraction of the peers instantaneously at zero cost. An equivalent scenario would be that, by threats or bribery, the adversary subverts the operators of that fraction of the peers. Thereafter, he has three strategies to choose from:

- The *nuisance* strategy attempts to waste time and resources at the unsubverted peers by raising alarms.
- The *stealth* strategy attempts to avoid detection while changing the consensus on the contents of the document, by causing polls to agree overwhelmingly on a faulty version of the document.
- The *attrition* strategy attempts to prevent peers from verifying their replica of the document long enough for random damage to corrupt it, by calling many spurious polls.

Frustrating the nuisance adversary requires both that the system have a very low natural rate of false alarms, and that the adversary exert large effort over a long period of time to cause an alarm. If it were merely enough that the effort be brief but large, or low but sustained, then the nuisance adversary with a flash worm or sufficient patience would succeed.

Frustrating the stealth adversary requires that biasing the sampling process take large efforts over long periods, and that the inherent intrusion detection be effective.

Frustrating the attrition adversary requires that it take large efforts over long periods of time for the adversary to consume enough resources at unsubverted peers to cause their polls to fail.

4. Analysis of techniques

We distill our experiences with the design of a peer-to-peer system that resists malicious attacks along three axes: *memory*, *effort*, and *autonomy*.

Memory is persistent state, such as reference lists, opinions about other peers, statistics on past local operation, etc. Peers use memory to understand, from their own vantage point, how the system evolves over time. However, reliance on large amounts of memory, or memory from long ago, is risky when a peer's own storage is unreliable, and when peer identities can be spoofed or subverted.

Effort is an instantaneous indication of a peer's willingness and ability to contribute to the well-being of the system. Peers may exact proofs of effort from each other in the process of a transaction; this can limit the transaction rate for both unsubverted and malign peers. MBF effort does not contribute directly to the well-being of the system and thus reduces its ability to deliver service. In our digital preservation context this is not significant.

Autonomy is a measure of how independent each peer is within the system. If autonomy is low, peer operations are determined by information from other peers or, in the limit, from a central controller. This can lead to sophisticated and efficient behaviors, but it allows an adversary capable of spoofing or subverting peers to have undue influence over the operations of unsubverted peers. If autonomy is high, peer operations are determined by local information, or in our case by local estimates of the consensus of the peer population. This skepticism about information from other peers limits both the ability of an adversary to control the behavior of unsubverted peers, yet also the ability of unsubverted peers to cooperate in reacting to exceptional conditions.

In the rest of this section, we describe the techniques we use in the LOCKSS system, we analyze how they fare on the three axes, and we provide reasoning for, and simulation evidence of, their usefulness. Overall, the nature of our preservation application allows us to trade effort for increased autonomy and moderate use of memory. We use the Narses [10] simulator.

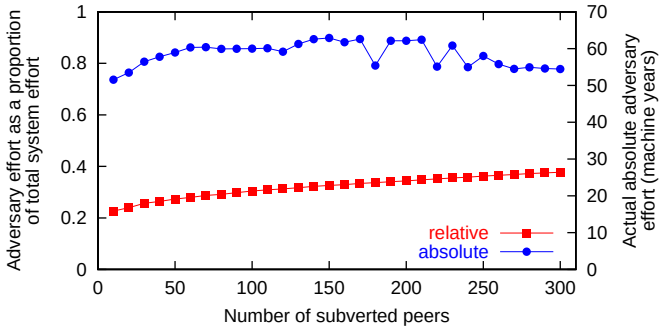

Fig. 1. Worst-case effort exerted by the stealth adversary capable of unlimited effort towards damaging the system in a network of 1000 peers, over 20 simulated years. On the left y axis, we graph the proportion of overall system effort exerted by the adversary ("relative" curve). On the right y axis, we graph the absolute effort exerted by the adversary ("absolute" curve).

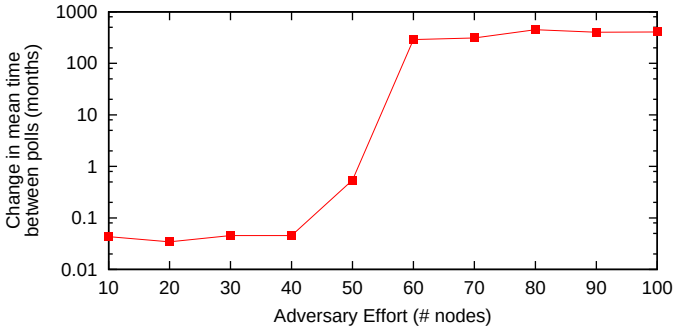

Fig. 2. Change in the mean time between LOCKSS polls, as a function of the computational power of the attrition adversary. The y axis is in logarithmic scale. There are 1000 peers in the system.

4.1 *Rate-limiting*

Designers should identify the maximum rate at which damage can be incurred in the system. This maximum rate is an important measure of the system's resilience against adversaries who can commit overwhelming resources to attacks, for short or long periods of time. Limiting this maximum damage rate is a powerful tool against attacks, since it can protect the system even from adversaries with unlimited resources.

LOCKSS peers can limit the rate of damage they incur because they decide autonomously when it is time to reevaluate their state, instead of watching the system around them for hints. As a result, an adversary who wishes to damage a peer through the protocol (as opposed to using physical, social, or legal means) must wait until that peer decides to reevaluate its state. Only then can the adversary apply his resources, even when those resources are unlimited, to sway the poll called by the peer and thereby affect the peer's document replica.

Figure 1 shows that the system limits adversaries capable of unlimited computational effort to applying only a limited amount of it. In a network of 1000 peers over 20 simulated years, the adversary never manages to exert more than 40% of the total system effort actually expended, which translates to no more than 65 machine years. And yet, even when the adversary initially subverts 30% of the peers, he can add no more than 5% to the initial 30% probability that a reader accesses an incorrect copy before detection [14].

Rate-limiting fares high on the autonomy axis but is independent of necessary effort or memory.

4.2 *Costly Signaling*

A peer must decide whether or not to admit other peers in a system operation: if the peer is calling a poll, it must decide which of the votes it receives to use; if the peer is invited to vote in somebody else's poll, it must decide whether to vote in that poll or not. The decision is important in both cases. In the former case, the peer reevaluates its local state based on the consensus of the votes it accepts; in the latter case, the peer commits significant resources in producing votes, so it must make sure its resources are not wasted.

LOCKSS peers use *costly signaling* to help make such decisions. Costly signaling is communication that carries a verifiable proof of effort along with protocol messages, based on the premise that a signal that was more expensive to produce must come from a sender who is good to do business with. In the field of biology, Gintis et al. [9] have shown that groups can evolve cooperation based solely on costly signaling under plausible conditions.

The type of verifiable effort in our costly signals is computation. In the MBF scheme we use [6], there are two system parameters: first, the system can set, almost arbitrarily, the cost of verifying effort; second, the system can set how much more expensive the cost of producing the effort is than verifying that same effort. By setting these parameters appropriately, we ensure that wasted effort caused by the attrition adversary is paid for with prior adversary investment. Similarly, we ensure that undetectably incorrect votes are expensive for the stealth adversary to produce.

Figure 2 graphs the change in the utility of our system, i.e., the change in mean time between polls, as a function of the computational power of the attrition adversary. The longer the time between polls, the longer random damage to a replica remains undetected. The attrition adversary can only increase the mean time between polls significantly if he commits 60 nodes or more to the attack. Using costly signaling this way we raise the barrier to an attrition attack, but not enough to be safe. Very preliminary simulations that take into account a node's time as well as

its effort in our signaling cost structure demonstrate more encouraging results.

Costly signaling requires high effort, but is independent of memory and of autonomy. LOCKSS peers use costly signaling as an option that permits high autonomy, as opposed to its alternative, third-party reputation.

4.3 Reputation

Reputation is a metric of the "trustworthiness" or quality of a peer, used by one peer to decide whether and how much to interact with another. It can be *first-person*, based solely on the local peer's experience, or *third-party*, based on testimony from other peers.

Third-party reputation is the result of peer collaboration to maintain the reputation database, collectively rewarding good behaviors and punishing bad behaviors as they are observed by subsets of the peer population. Third-party reputation, as a special case of peer state, is placed high on the memory axis. It is low on the autonomy axis, however, because it is accumulated over transactions that do not necessarily involve the peer using it.

To move higher on the autonomy axis and avoid relying on the testimony of others, we focus on first-person reputation. LOCKSS peers rely only on their own experience, verifying costly signals from other peers, to populate their reference lists. This prevents malign peers from cheaply inflating their own reputations or degrading the reputations of unsubverted peers, but this approach can suffer in systems where there is a low probability of repeat transactions between any two peers.

4.4 Expiration of Memory

Adversaries can try to populate the reference lists of unsubverted peers with malign peers to achieve their goals. By doing so, the nuisance adversary can cause an alarm to be raised faster, and the stealth adversary can damage more documents for longer before detection.

To prevent this, entries in the reference lists time out and are evicted after a few polls. Similarly, after a peer calls a poll, it removes from its reference list those peers that participated in the poll. This approach is particularly effective against prudent adversaries with limited power, who manage their resources so as to launch the most cost-effective attack. As a result, an adversary must invest sustained effort for as long as he wishes to maintain a presence in unsubverted peers' reference lists.

Because memory expiration limits the lifetime of a peer's observations, we place the technique at the low end of the memory axis. Because memory expiration results in higher and sustained effort requirements for the system, we place the technique at the high end of the effort axis.

4.5 A Regulated Economy

We believe a successful system cannot have an unregulated economy. It is unrealistic to expect a completely autonomous, unregulated system to be able to heal itself always under all circumstances without external input. A system may achieve an equilibrium during normal operation, but it is rare for exceptional activities, in our case an attack based on a common vulnerability among a fraction of the peers, to be handled purely within the system. In such cases (fraud in the real world), the intervention of an external agent (law enforcement agencies and the courts in the real world) is necessary to resolve the problem and restore normal operation.

LOCKSS peers use *alarms* as indicators of exceptional conditions in the system. A peer raises an alarm when 1) it detects coherent damage in the system, 2) it suspects tampering with its local network or other resources, or 3) it has been unable to participate in the system for a time inconsistent with transient network failures.

Alone or in concert, human operators respond to alarms by identifying the problem using the forensic information the system collects and by restoring normal operation. We agree with conventional intrusion detection systems that attacks on a system can only be repelled by cooperation between the software and the humans responsible for it.

We classify our kind of regulated economy as highly, but not completely, autonomous, because for the first alarm type, more than a few peers must interact to thwart the attempts of the adversary. Because the latter two alarm types require temporarily storing observations from multiple system operations, we place this technique in the middle of the memory axis.

5. Research Agenda

Our investigations into the effectiveness of economic measures at resisting attacks on the LOCKSS system are encouraging. In simulations, the system resists several kinds of attack from a very powerful adversary. For example, with an initial subversion of 30% of the peers, an unlimited adversary increases the worst-case probability that a reader will access an incorrect copy by only 5%. However, there is much more to do before we understand the limits of our techniques or how best to apply them in practice. The primary topics of the LOCKSS research agenda include better adversary strategies, the use of first-person reputation to thwart selfish peers, improved attrition resistance, and trade-offs between memory and effort.

We have not yet shown our adversary strategies are optimal from his perspective. We need to explore other strategies, measuring the degradation they produce for

given time and effort invested by the adversary, and investigate ways to prove that an adversary strategy is optimal.

We also believe that remembering first-person reputation for longer could help us tackle certain selfish behaviors. For example, although a peer cannot obtain a repair for a damaged document without having participated in other peers' recent polls, the LOCKSS protocol is currently vulnerable to peers who participate correctly in polls but never supply repairs to others. We could penalize the reputation of such selfish peers.

Using costly signaling, we have raised the barrier to an attrition attack, but not enough to be safe. A LOCKSS poll begins with a three-way handshake: poll invitation, followed by a challenge from the invitee, followed by a proof of effort from the poll initiator. Currently, construction of the initial poll invitation message is cheap. As a result, a malicious adversary can send such a message, causing a peer to respond and then wait for an effort proof that might never come. By taking this time into account, in addition to effort, in the design of our signaling cost structure, we can reduce this kind of cheap attack path open to the adversary. Extending our simulation to include these costs has produced encouraging initial results.

Another approach to thwarting the attrition adversary is to combine costly signaling with first-person reputation. Currently, the signal required of a peer initiating a poll is the same regardless of whether or not that peer has had any previous interaction with the invited peers. For applications where repeat transactions are frequent, designating all recently-unseen peers as *potentially* uncooperative can perhaps limit the adversary in a manner similar to the make-newcomers-pay strategy [8]. According to this strategy, LOCKSS peers would require a higher payment (i.e., costlier signal) for services from those for whom they have no recent opinion than for those for whom they have a positive recent opinion. This is an example of the need to combine reputation and costly signaling; neither is adequate independently. There may well be other areas in which this synergy could be exploited.

The trade-off between memory and recent effort deserves investigation. Accumulating memory over several polls may allow peers to detect long-term attacks or selfish behavior. It would also allow peers to make better decisions about where to deploy their limited resources. But the longer the period over which this history is accumulated, the greater the vulnerability of the system to sudden subversion of previously unsubverted peers, or sudden changes of tactics by the malign peers. Proof of recent effort instead helps us reduce these vulnerabilities.

Finally, we hope to migrate our techniques to the deployed LOCKSS system over the course of the next year. As we do so, we will surely learn more about the practicality of our ideas for resisting attack in real systems.

Acknowledgments: This work is supported by the NSF (Grant No. 9907296), by Sun Microsystems Laboratories, by DARPA (contract No. N66001-00-C-8015), and by MURI (award No. F49620-00-1-0330).

REFERENCES

[1] M. Abadi, M. Burrows, M. Manasse, and T. Wobber. Moderately Hard, Memory-bound Functions. In NDSS, 2003.

[2] R. Anderson. The Eternity Service. In PRAGOCRYPT, 1996.

[3] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal Multicast. *ACM TOCS*, 17(2):41–88, 1999.

[4] W. Diffie. Perspective: Decrypting The Secret to Strong Security. `http://news.com.com/2010-1071-980462.html`, Jan 2003.

[5] R. Dingledine, M. Freedman, and D. Molnar. The Free Haven Project: Distributed Anonymous Storage Service. In Hannes Federrath, editor, *Proc. of the Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[6] C. Dwork, A. Goldberg, and M. Naor. On Memory-Bound Functions for Fighting Spam. In *CRYPTO*, 2003. To appear.

[7] M. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Layer. In *CCS*, 2002.

[8] E. Friedman and P. Resnick. The Social Costs of Cheap Pseudonyms. *Journal of Economics and Management Strategy*, 10(2):173–199, Summer 2001.

[9] H. Gintis, E.A. Smith, and S. Bowles. Costly Signalling and Cooperation. *Journal of Theoretical Biololgy*, 213:103–119, 2001.

[10] TJ Giuli and M. Baker. Narses: A Scalable, Flow-Based Network Simulator. Technical Report cs.PF/0211024, Computer Science Department, Stanford University, Stanford, CA, USA, Nov 2002.

[11] A. Goldberg and P. Yianilos. Towards an Archival Intermemory. In *Proceedings of IEEE ADL*, 1998.

[12] M. Keller, V. Reich, and A. Herkovic. What is a Library Anymore, Anyway? *First Monday*, 8(5), May 2003. `http://www.firstmonday.org/issues/issue8_5/keller/index.html`.

[13] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An Architecture for Global-Scale Persistent Storage. In *ASPLOS*, 2000.

[14] P. Maniatis, M. Roussopoulos, TJ Giuli, D. S. H. Rosenthal, M. Baker, and Y. Muliadi. The LOCKSS Opinion Polling Protocol, Mar 2003. Under submission.

[15] V. Reich. Stanford Libraries. Personal Communication.

[16] M. Rennhard. Practical Anonymity for the Masses with Mix-Networks. Technical Report TIK-Nr. 157, Swiss Federal Institute of Technology, Feb 2003.

[17] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *Proceedings of the Workshop on Privacy in the Electronic Society (in association with CCS)*, Nov 2002.

[18] D. S. H. Rosenthal and V. Reich. Permanent Web Publishing. In *Freenix*, 2000.

[19] D. Spinellis. The Decay and Failures of Web References. *Communications of the ACM*, 46(1):71–77, Jan 2003.

[20] S. Staniford, V. Paxson, and N. Weaver. How to 0wn the Internet in Your Spare Time. In *USENIX Security Symposium*, 2002.

[21] N. Wingfield. EBay's Figurine Scandal: Auction Site Merchant Disappears With The Goods. Wall Street Journal, 2/22/2002.