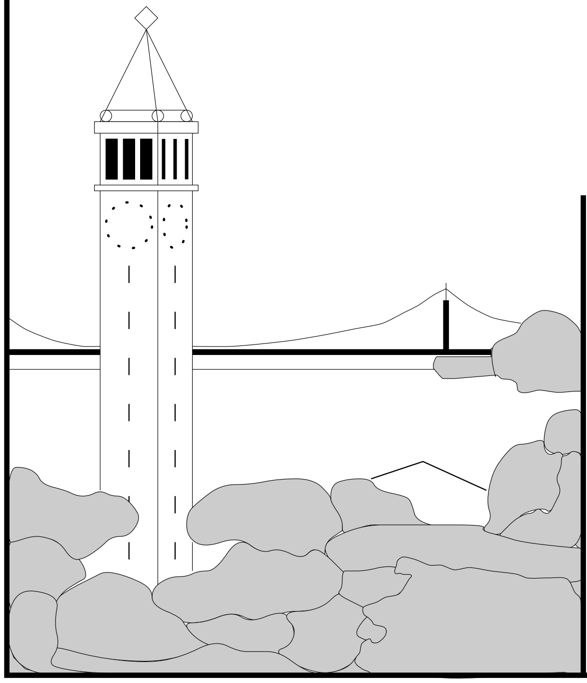
Anonymity in Structured Peer-to-Peer Networks

Nikita Borisov and Jason Waddle



Report No. UCB/CSD-05-1390

May 2005

Computer Science Division (EECS)
University of California
Berkeley, California 94720

Anonymity in Structured Peer-to-Peer Networks

Nikita Borisov and Jason Waddle

May 2005

Abstract

Existing peer-to-peer systems that aim to provide anonymity to its users are based on networks with unstructured or loosely-structured routing algorithms. Structured routing offers performance and robustness guarantees that these systems are unable to achieve. We therefore investigate adding anonymity support to structured peer-to-peer networks. We apply an entropy-based anonymity metric to Chord [SMK$^+$01] and use this metric to quantify the improvements in anonymity afforded by several possible extensions. We identify particular properties of Chord that have the strongest effect on anonymity and propose a routing extension that allows a general trade-off between anonymity and performance. Our results should be applicable to other structured peer-to-peer systems.

1 Introduction

Preserving the privacy of computer users is a growing concern. In recent years we have seen various organizations make frequent demands for the records of users' activity, in the name of defending themselves from piracy, whistle-blowing, trade secret leaks, and so on. While some of the demands are supported by judicial review, others are obtained through intimidation, opening up avenues for abuse.

There have been several efforts to deploy privacy enhancing, and specifically anonymity, technologies on the Internet. Most of the robust technologies rely on forming an overlay network of nodes to forward messages such that the identity of the originator is masked. However, most systems have suffered from low popularity; this is detrimental to the systems' success, as the ability to preserve privacy relies on having a large number of participants.

Scalable peer-to-peer networks are a promising platform for anonymity technologies: the systems already contain mechanisms for forming and maintaining overlay networks, and the benefits of peer-to-peer technologies have been responsible for rapid adoption of several peer-to-peer systems on the Internet, sometimes reaching user populations of several million.

Several existing anonymous systems use a scalable peer-to-peer design [BG03, CSWH00]; however, they rely on a form of restricted flooding for routing. Recent research in the peer-to-peer community has focused on *structured* peer-to-peer networks, built around invariants that are able to guarantee bounds on path length, state management overhead, number of inter-node connections, etc., and ensure delivery to any node in a network. In this paper, we study the feasibility of using such structured networks to build anonymous systems.

Most of the structured networks implement an abstract interface called a *distributed hash table*; we thus focus on performing the basic operations of this abstraction — "get" and "put" — anonymously. Further, each operation involves two endpoints: the client requesting the operation and the node actually performing the hash table lookup or store. The location of the latter endpoint is highly restricted by the structure of the network and is therefore hard to anonymize; therefore, we study how to hide the initiator of the operation, or so-called *source-anonymity*.

We base our study on the Chord [SMK$^+$01] network; however, many of our results should be applicable to other types of structured networks as they rely on graph properties of the Chord network that exist in other systems as well. We perform our analysis using an information-theoretic entropy metric, first applying it to the unmodified version of Chord routing and then to several potential modifications. We also attempt to identify the variables that affect the anonymity metric and use them to form general conclusions about anonymity in structured networks.

The rest of the paper is organized as follows. Section 2 reviews other relevant research in anonymous systems; Section 3 explains the anonymity metric that we use and our methodology. In Section 4 we perform a qualitative analysis of the factors behind anonymity in Chord, as well as a simple modification using randomized choice. Section 5 performs a quantitative comparison of several potential routing extensions. In Section 6 we explore a modification to the underlying Chord structure, enable a new anonymous routing algorithm avoiding some of the pitfalls common to the methods that use the regular Chord network. Section 7 discusses some further research directions, and Section 8 concludes.

2 Related Work

Preserving anonymity has been a topic of much study in the literature. One of the earliest proposals for how to achieve anonymity was Chaum's Mixes [Cha81]. It suggested forwarding messages through a chain of nodes, using layered encryption to ensure that each node can only know its predecessor and successor in the chain. This idea was used in many anonymous routing designs [Dai96, STRL00, BGS01, FM02], several of which were deployed and used to route Internet traffic. However, the mix system requires every mix node to send cover traffic to every other mix node, limiting scalability to few (usually under 100) nodes.

This limitation comes from the fact that mixes are designed to resist a very powerful adversary who is able to monitor every link in the network and perform traffic analysis to correlate messages. Other systems have relaxed this assumption, limiting the attacker to monitoring only a subset of links. MorphMix [RP02] justifies this assumption by supporting networks of very large number of nodes, where a global adversary is infeasible. To enable such scaling, each node only keeps track of a limited number of nodes and relies on its neighbors to discover other nodes to use for building mix chains. This method becomes ineffective when some of the neighbors are attackers; to avoid this problem, MorphMix uses a complicated scheme to detect colluding attackers and also constantly evolves the state of the network to limit exposure. Some other work restricted-topology mix networks has been on Mix Cascades [DS02], designed to resist very powerful adversaries who control all but one mix nodes, and the use of expander graphs [Dan03] to reduce the amount of cover traffic while stile maintaining some anonymity guarantees.

Another focus of research is on anonymous publishing and reading of data, rather than anonymous routing. The anonymous publication problem has attracted the attention of the file sharing community; the designers of Gnutella [HB] aimed to provide anonymity, though their protocol anonymizes only queries and not data transfers. [Cle] Freenet [CSWH00] and GNUnet [BG03] avoid this problem by forwarding data transfers through the network, the same as queries and responses. None of these networks have a rigid structure and therefore employ some form of flooding (directed in the case of Freenet) to route queries.

There has been some study into building anonymity into structured peer-to-peer networks; AChord [HW02] proposes a modification to Chord that keeps the location of a predecessor node for an ID secret, except from its neighbors in the Chord ring. Kumar and Bansal [KB02] studied modifications to Chord routing to improve source anonymity; their work is the most closely related to ours. However, their analysis was based on probability of identification and not the information-theoretic metric we used.

We discuss the use of metrics in comparative studies of systems in the following section. Another method of formal analysis of anonymity systems is defined by the predecessor attack [WALS02, WALS03], which explains how anonymity of a system degrades over time. We do not analyze our modifications to Chord routing using the predecessor attack, but rather leave that as future work; One reason for this is that the results of the predecessor attack must be interpreted in the context of how often different paths are chosen to route to the same destination; this context would only become more apparent in a more concrete system design.

3 Measuring Anonymity

We aim to employ rigorous analysis as much as possible in our explorations of anonymity. In taking this approach, we need to develop models of the systems we analyze, and the usual tension arises: the models must be simultaneously simple enough to analyze, general enough to apply broadly, and detailed enough to capture all relevant factors. Here we discuss and give some justification for our choice of model.

3.1 General Model

Situations where anonymity is relevant can usually be framed in an adversarial manner: there is an *attacker* who wishes to learn the identity of a particular participant among a crowd Ω of participants.

We choose to sacrifice a little bit of generality by focusing on *source anonymity*. That is, the attacker attempts to learn the identity of the participant based on some active behavior on the part of the participant. Thus, in referring to this particular participant, we will call her the *initiator*, and her relevant behavior an *event*.

The attacker observes some information that is probabilistically related to the event, and from that information, possibly learns something about the identity of the initiator. Our concept of anonymity is exactly the attacker's *uncertainty* with respect to the identity of the initiator of the event based on the observation.

Notice that we allow the attacker to actively participate in the process of identifying the initiator (insofar as the particular model allows the attacker to influence the system), but we do not concern ourselves with what the attacker can learn in the absence of an event caused by an initiator. In concrete terms, we do allow an attacker who also participates in a network to deviate from the usual routing protocol and make routing decisions that may help him learn the initiator's identity, but we do not care that an attacker may learn the identity of whoever is responsible for a particular location in the hash table (thus, we do not consider *destination anonymity*, since a destination has basically a passive role).

Finally, we note that this model can handle multiple events by treating them as a single product event.

3.1.1 Definitions and Notation

Formally, we have a random variables S and F.[1] The initiator (or source) is indicated by S, and is distributed uniformly on Ω, so $\Pr[S = s] = \frac{1}{|\Omega|}$ for all $s \in \Omega$.

We assume the source initiates some event, and the attacker observes some effect of this event; the set of possible observations is $\mathcal{F}$, and the random variable F indicates which effect is observed. The two random variables have a joint distribution $p : \Omega \times \mathcal{F} \to [0, 1)$, where $p(s, f) \equiv \Pr[S = s \ \wedge \ F = f]$. We overload this notation by referring to the marginal distributions $p : \Omega \to [0, 1)$ and $p : \mathcal{F} \to [0, 1)$.

Intuitively, the correlation of these two random variables S and F is related to how much the attacker learns about the identity of the initiator. We now must decide how to quantify this correlation in a way that helps us determine the overall anonymity-preserving properties of a system.

3.2 Anonymity Sets

Anonymity sets capture the intuitive notion of blending in with a crowd. The idea is that the attacker learns about the initiator's identity by using his observed information to rule out other potential initiators.

3.2.1 Definition of the Anonymity Set Metric

The anonymity set is the subset of Ω that contains exactly those who could have possibly initiated the event observed by the attacker. In our notation, if the attacker observes outcome $F = f$, then the anonymity set is $\{s \in \Omega \mid p(s, f) > 0\}$. We let A be the random variable indicating the anonymity set deduced by the attacker from observing F.

To interpret the anonymity set as a metric, we consider the expected size of the anonymity set $\mathrm{E}[|A|]$ and normalize that by the best possible value, $|\Omega|$. Using anonymity sets as a metric, one would then quantify the anonymity of a system by computing:

$$\frac{\mathrm{E}[|A|]}{|\Omega|},$$

where the score ranges from $\frac{1}{|\Omega|}$ (guaranteed identification) to 1 (guaranteed perfect anonymity).

3.2.2 Example

Suppose the attacker stands blindfolded in the middle of a room with $|\Omega| = 10$ other people: 5 to his left and 5 to his right. The initiator, chosen at random, snaps her fingers once. The attacker observes the direction of the sound and nothing else, so $F = \{\text{LEFT}, \text{RIGHT}\}$ and F is perfectly correlated with which half of the room the initiator is in. The anonymity set metric assigns to this system a value of:

$$\frac{\mathrm{E}[|A|]}{|\Omega|} = \frac{5}{10} = \frac{1}{2}.$$

[1]Our choice of F as a name of this random variable comes from our analysis of Chord. The reason will be clear in later sections.

3.2.3 Problems with Anonymity Sets

There are two major problems with the anonymity set metric. One is that it is defined in terms of *expectation*. On its own, all it gives is the average anonymity while saying nothing about variance. Thus, a middle-of-the-road score 1/2 could either mean that participants always retain half their anonymity or that they are completely identified roughly half of the time and remain completely anonymous otherwise. A guaranteed degree of less anonymity (but better than certain identification) might be psychologically more satisfying to the participants.

Another serious problem stems form the definition of the anonymity set, on which the metric is based. By including all of Ω that could have *possibly* induced the observed effect, we ignore that the attacker may have enough information to identify the the initiator (or at least a smaller set of possible initiators) with high probability.

For example, suppose $\Omega = \{0, \ldots, 9\}$ and $\mathcal{F} = \{f_0, \ldots, f_9\}$, and that:

$$p(s, f_y) = \begin{cases} .091 & \text{if } s = y, \\ .001 & \text{if } s \neq y. \end{cases}$$

This implies the conditional distributions

$$p(s|f_y) = \begin{cases} .91 & \text{if } s = y, \\ .01 & \text{if } s \neq y. \end{cases}$$

.

Thus the attacker, after observing $F = F_y$, can be 91% certain that y was the initiator. All other initiators are possible, however, so $A = \Omega$ with probability 1, and the anonymity set metric dubiously assigns the value 1 to this system.

3.3 Entropy

Anonymity sets then fail because they do not consider attackers that have a probabilistic advantage in identifying initiators. We need to employ some measurement that also quantifies the difference between the induced distribution and the ideal uniform distribution. This need is well-acknowledged, and multiple researchers [DSCP02, SD02] have independently come up with the approach discussed here.

3.3.1 Definition of the Entropy Metric

Observing that anonymity sets are too coarse for the analysis of systems that frequently result in non-uniform conditional distributions $\Pr[S = s|F = f]$, it is natural to consider the expected *entropy* or the distribution on S given F. This quantity is known as the *conditional entropy* and is denoted $H(S|F)$. The formula for condition entropy is:

$$H(S|F) \equiv \sum_{f \in \mathcal{F}} \Pr[F = f] H(S|F = f),$$

where the quantity $H(S|F = f)$ is defined as:

$$\begin{aligned} H(S|F = f) &\equiv -\sum_{s \in \Omega} \Pr[S = s|F = f] \log_2 \Pr[S = s|F = f] \\ &= -\sum_{s \in \Omega} \frac{p(s, f)}{p(f)} \log_2 \frac{p(s, f)}{p(f)}. \end{aligned}$$

Although the formula is more complicated, the interpretation is almost as simple as that of the anonymity set. It answers the question: *On average, how many more bits of information does the attacker need to identify an initiator?* or, alternatively, *On average, how many bits of anonymity does an initiator give up?*

As in the anonymity set case, in order to compare systems with various sizes of $|\Omega|$, we define the entropy anonymity metric as the conditional entropy normalized by the best possible value, $\log_2 |\Omega|$. Then the entropy metric for a particular system is calculated:

$$\frac{H(S|F)}{\log_2 |\Omega|},$$

where the values range from 0 (guaranteed identification) to 1 (guaranteed perfect anonymity).

3.3.2 Example

Consider again the problematic example from Section 3.2. While the anonymity set metric gave a value 1 to this system, the entropy metric gives:

$$\begin{aligned}\frac{H(S|F)}{\log_2|\Omega|} &= -\left(.91\log_2 .91 + .09\log_2 .01\right)/\log_2 10\\ &\approx .1314,\end{aligned}$$

with the interpretation that, on average, only about 13% of the bits of the initiator's identity are kept hidden from the attacker.

3.3.3 Problems with the Entropy Metric

The entropy metric shares the problem with the anonymity set that it is based on expectation: a particular expected entropy could either mean that all induced distributions are good or that some are really good and others are really bad.

In addition, conditional entropy can be somewhat harder to analyze than anonymity sets. In a large system, computing the probability of the distribution of events given an observation requires analysis of a large number of random variables; non-uniform distribution of these variables greatly complicates the analysis. The metrics have been most successfully applied to to fully symmetric networks, such as Onion Routing [STRL00] and Crowds [RR98], while complicated networks like Freenet [CSWH00] have thus far resisted analysis. In this paper, we develop an empirical, simulation-based methodology to overcome such difficulties.

3.4 Empirical Entropy Measurement

If we were able to compute the probability $P(S = s|F = f)$ for every pair of events and observations (s, f), we would be able to compute the entropy metric following its defining formula. We can use a simulation of a system to estimate these probabilities. Given a state of a system, we can generate an event s and then simulate the system, noting all the relevant attacker observations f in a log: $L = L \cup (s, f)$. If we repeat this process for every possible even f, we can proceed to estimate $P(S = s|F = f)$ using the log:

$$P(S = s|F = f) \approx \frac{|\{(e, o) \in L|e = s, o = f\}|}{|\{(e, o) \in L|o = f\}|}$$

Notice that if the system is deterministic, our computation will be exact: if s results in f, the probability will be $1/|A|$ where A is the anonymity set corresponding to f. Otherwise, the probability will be 0. In a non-deterministic system, the value we compute will depend on the choices made in the simulation. We can improve the estimate by iterating this process, generating an event s multiple times and each time recording the observations that got made in the log. We need to change the computation to use multisets, but otherwise the same formula applies. In effect, we are sampling the distribution $P(S|F = f)$, so as the number of iterations increases, our estimate will become more precise.

The main disadvantage of this approach is that the computed distribution is dependent on an initial state of the system. It may be the case that other initial states produce different distributions. We can check whether this is the case by testing several states, and resort to computing the average distribution based on the states. Also, if there is a strong dependence on the initial state, the entropy metric will overestimate the information available to the attacker. The probability calculated is actually $P(S = s|F = f, I = i)$ where i is the initial state. If the attacker does not know i, he may not be able to turn an observation f into an attack, even if the computed entropy is very low.

A lesser problem is that the distribution is sampled, rather than computed. We argue that in most cases, this will also lead to the entropy metric being conservative, as sampling will tend to overestimate high-probability figures and underestimate low-probability ones. (In many of our tests, we used an extension that computes exact probabilities rather than sampling; we found that in most cases, there was no significant difference in the computed entropies using two methods.) Nonetheless, the empirical metric is useful, as it is a rigorous, if conservative, estimate that can be applied to many complex systems. Most importantly, it can account for attacks not thought of in informal analysis.

4 Anonymity in Chord

Having defined the anonymity metric and our methodology for computing it, we proceed to apply this metric to Chord [SMK$^+$01] routing and a randomized modification. The results will be used to motivate the choice of other routing algorithms we will study below.

4.1 Routing Algorithm

The unmodified version Chord routing can be used to provide some amount of source-anonymity if we use *recursive* routing and avoid explicitly revealing the source. In recursive routing, each intermediate node has the responsibility of forwarding the message to the next node along the route; in contrast, in *iterative* routing the originator contacts successive nodes along the route in order to determine who the next node should be, eventually reaching the destination and sending the message to it directly. [SM02] While iterative routing is clearly incompatible with source-anonymity, recursive routing can help hide the identity of the sender, given that the identity is not included in the message itself. If a response to a message is desired, it must be forwarded back along the routing path; in the simplest version, this can be accomplished by each node keeping a record of messages it has routed and from which node they arrived, indexed by message or flow identifiers. This method is susceptible to denial of service attacks, since each message sent requires each forwarder to keep state about the message, thus an attacker could cause a node to use up all of its resources by sending messages. One way to avoid such problems is to store the state in the message itself and include it in the response, using encryption to avoid revealing the entire path to the destination.

4.2 Measuring Anonymity

We will begin our analysis by measuring the entropy achieved by such routing in Chord. We built a simulator that constructs a random Chord network with N nodes and make each node an attacker with probability c. It then simulates the path that a message sent from an arbitrary node would take being routed to destination of 0.0. (Since the network is randomly generated, we can fix this arbitrary destination without loss of generality). If a message hits an attacker node, we record which finger was used to arrive there, and stop routing the message further. In a real network, attackers may wish to still faithfully forward incoming messages in order to avoid detection; however, doing so reveals no new information to the group of attackers, therefore we drop the messages in our model. After calculating the paths from each honest node, we use the attacker logs to compute the conditional probability distribution of the sources given the observed event of a message arriving at a finger (in this case, $1/k$ for each node mentioned in the log, and 0 for others, where k is the number of records in the log for a finger). This probability distribution is then used to compute the conditional entropy metric, as described in the previous section, representing the expected amount of uncertainty the attacker has after a single message is sent.

Figure 1 plots the results for various values of n and c. As we expect, the entropy metric falls as the number of attackers decreases. One trend we can notice is that the *proportion* of attackers (c) is better correlated with the entropy than the absolute number (Nc). We consider this a positive result, since as a network grows in popularity, the number of cooperating attackers necessary to achieve equivalent results grows linearly. Another thing to notice is that the variance of the measured entropy (computed over 20 runs) is quite high. To find out the source of this variance, we built a visualization tool to track the paths taken by messages and the information gathered by attackers.

4.3 Explaining Variance

The tool, shown in Figure 2, displays the Chord nodes arranged in a circle according to their ID. To aid visualization, we only ran the tool on networks of size 128; the data from Figure 1 suggests that even this small network size is reasonably representative. One immediately apparent effect is that the nodes are not evenly distributed along the circle; there are some clusters of many nodes tightly packed together and some large gaps containing no nodes. This distribution affects the *indegree* of nodes, i.e. the number of fingers pointing to a node: nodes within a cluster have few incoming fingers, while nodes following a large gap have many. In our networks of 128 nodes, the indegree of some nodes was only 1 node (the successor link required by Chord), while some other nodes had an indegree of over 32. Intuitively, an attacker with a large indegree is going to have more impact on the network than one with incoming links, because it can see more messages.

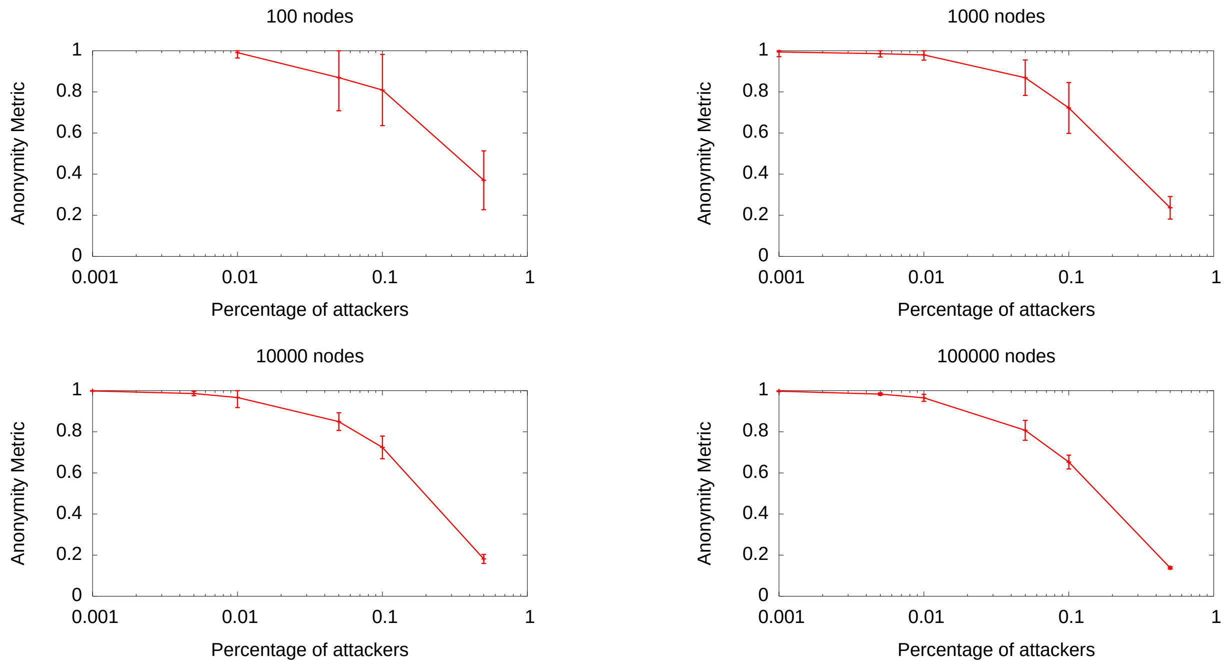


Figure 1: Anonymity Results for 100, 1000, 10000, and 100000 node networks.

4.4 Effects of Attacker Placement

To confirm this intuition, we had our tool color the attacker nodes based on how much information each attacker was providing. We approximate the information from each attacker by the following formula:

$$I(A) = \sum_{f \in \mathrm{Fingers}(A)} P(F=f)H(S|F=f) + \left(1 - \sum_{f \in \mathrm{Fingers}(A)} P(F=f)\right)$$

where $\mathrm{Fingers}(A)$ is the set of fingers that points to attacker A. The formula approximates the entropy metric based on the information only from attacker A,[2] and is closely related to the per-attacker component in the total entropy metric calculation. We did, in fact, observe that attackers with more information tended to have a larger indegree. We also noticed another effect, namely that attackers closer closer to the destination were contributing more information than those further away. Once again, this is because attackers closer to the destination are able to intercept more messages.

These observations show the importance of attacker placement. We ran some simulations where attackers were placed such that they would have the largest possible indegree, and would be closer to the destination. We observed that picking only 10 attackers out of 1000 in this way produced an average entropy metric of 0.84, which is worse than 50 randomly chosen attackers. Picking the worst 100 attackers produced an average of 0.33, more than 3 standard deviations away from the mean obtained with 100 random attackers! This shows that if attackers are able to choose their place in the network, they can dramatically decrease the achieved anonymity. Hence, schemes limiting the freedom of the attackers to choose their position in the network, such as tying it to their IP address, can be of great help when trying to design anonymous routing.

4.4.1 Honest Node Variance

Just as not all attackers contribute the same amount of information, not all honest nodes are able to achieve the same level of anonymity. Some nodes' messages reach the destination without ever getting intercepted, others will attempt

[2]The reason it is an approximation is that an observed incoming message at A implies that the message did not take a path through another attacker A', as the probability distribution discounts such paths.

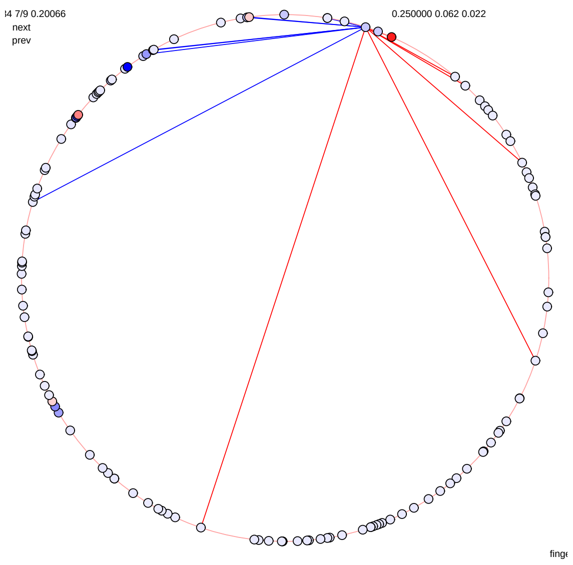


Figure 2: Visualization Tool.

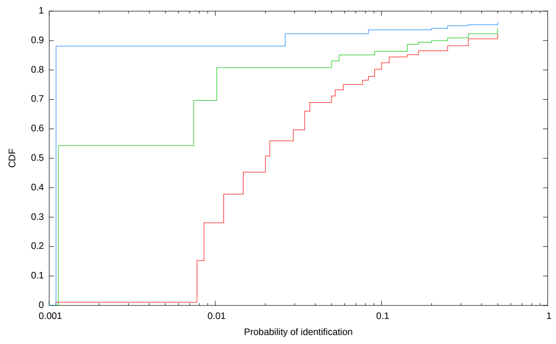


Figure 3: CDF of the probability of identification for 3 different 1000-node networks.

to route through an attacker mid-way on a path, and others still might chose an attacker as their first hop. We would like to compute the per-node exposure for a given network scenario; the formula we use is the *probability of identification*:

$$P_{\text{ident}}(s) = \sum_f P(S=s, F=f)P(S=s|F=f) + \frac{1}{|S|}\left(1 - \sum_f P(S=s, F=f)\right)$$

The first term in the formula is the expected probability the attackers assign to the true source whenever the source's message reaches an attacker, and the second represents a random guess on the part of the attackers as to the source in the case the message is not intercepted. (This term ensures that $P_{\text{ident}}(s) \geq 1/|S|$.)

Figure 3 plots the cumulative distribution of the probability of identification of honest nodes for 3 different networks of size 1000, with 100 attackers. In the case of the top two lines, the majority of the nodes has a very low probability of identification, as their messages do not get intercepted. However, other nodes have a significantly higher probability of identification, with several at the extreme of 0.5. The bottom plot shows a network where several attackers are close to the destination and intercept most of the messages.

What causes this irregularity? It turns out that the same network properties that are responsible for the variance in attacker nodes are at play here. The nodes with the highest probability of identification are those with the lowest indegree and those furthest away from the destination. In other words, the same situation that is good for an attacker is good for an honest node. This can be explained by the fact that a node achieves anonymity through forwarding the messages of other nodes, by hiding its own messages among those it forwards. [BG03] A node can maintain anonymity in this way even if all its fingers point to attackers. That being said, having attackers for neighbors is detrimental to anonymity in general, because there is less of a chance of your messages getting further mixed with those of nodes closer to the destination.

4.5 Randomized Routing

So far we have looked at results for the standard routing algorithm of Chord. A primary feature of this algorithm is that it is deterministic: the path taken by a message is completely determined by the destination Chord. When we

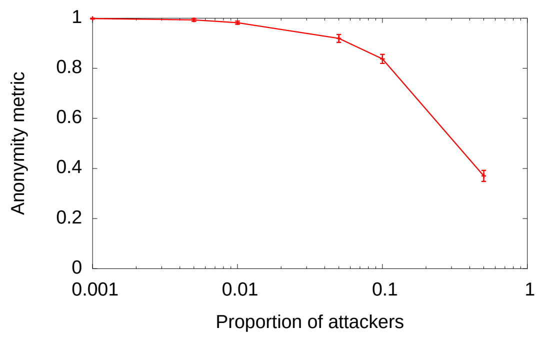


Figure 4: Entropy metric for random routing in a 1000-node network.

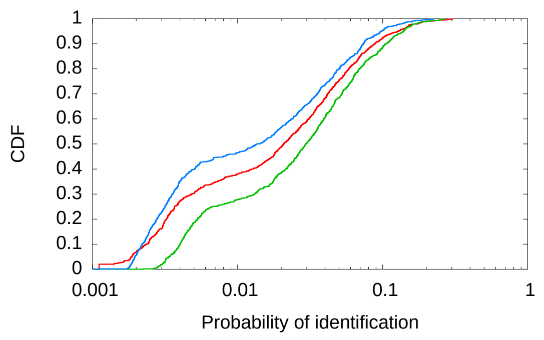


Figure 5: Cumulative distribution of probability of identification for random routing in a 1000-node network.

Routing	**Dist**	**Indegree**	**Num Attackers**
Chord	0.01	0.004	0.11
Random	0.09	0.09	0.40
Regular	0.30	0	0.46

Table 1: Correlations of potential factors influencing probability of identification.

look at some possible modifications to the routing algorithm, we will add some random choices to the path selection. Before we compare such algorithms, we would like to know how non-determinism affects the anonymity behavior of a network. To answer this question, we study one of the simpler random routing algorithms.

In standard Chord routing, to forward a message a node chooses the longest finger that brings it closer to the destination. In our random routing modification, we relax this requirement and choose *any* finger, as long as that finger ensures progress towards the destination. A message is still guaranteed to arrive at the destination eventually, but the path to get there can be significantly longer. This algorithm represents the least restrictive choice of routes while staying within the original Chord network structure and maintaining the invariant of making progress at each step.

Figure 4 plots the entropy for random routing in a 1000-node network. Compared to the graph in Figure 1, we notice not only that the entropy metric is higher, but that the variance is much smaller. We must conclude that some of the effects that caused the variance in the static routing case must be diminished. In random routing, there are many more possible paths that a message may take through the network. In particular, both high- and low-indegree nodes can potentially route messages from any preceding node, and hence the disparity between the nodes is diminished. The increase of the size of the anonymity set is reflected in higher values in the entropy metric.

Figure 5 plots the cumulative distribution of the probability of identification for honest nodes, for three different network scenarios. Unlike Figure 3, the three lines are much more similar; we can also see that the nodes are much more evenly distributed in the CDF. Once again, randomization seems to move nodes away from the extremes: few nodes have the minimal probability of identification, but as a result, few nodes have a really high probability either.

However, our experiments with the visualization tool show that the indegree and the position in the network still affect the probability of identification. In fact, these factors prove to be *better* indicators of the exposure of a node than in the deterministic routing case. Table 1 shows the correlation coefficient of a least squares fitting relating the factors to the probability of identification, for both the random and deterministic routing cases. We also computed the correlation coefficients for random routing on a “regular” Chord network, where all of the nodes are evenly spaced and observed that the distance correlation is much stronger in this case. We conjecture that there must be some secondary effects of irregular network structure that are not captured by the indegree computation which are responsible for some of the variance seen in the random routing; perhaps relating to the shape of the network connection graph of nodes closer to the destination than the node. This last conjecture is supported by the fact that the number of fingers that point to attackers, shown in the last column, is the strongest indicator of the probability of identification for all cases. If a finger points to an attacker, the structure of the network beyond that becomes irrelevant as the message will not be

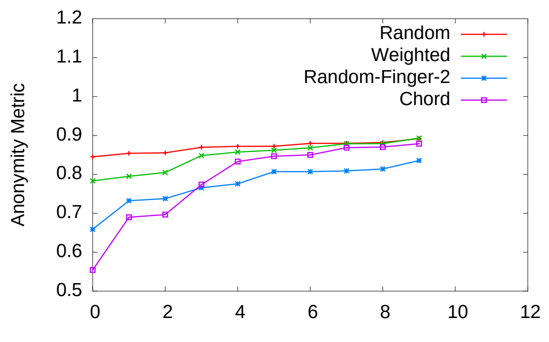


Figure 6: Comparison of routing algorithms.

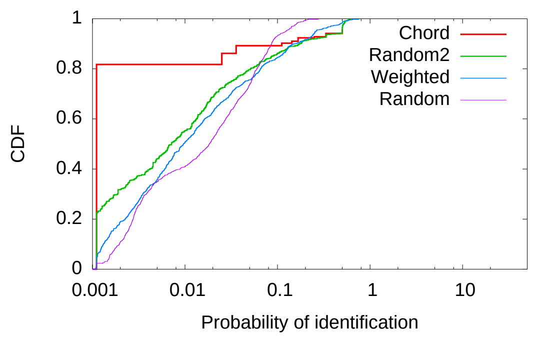


Figure 7: Probability of identification for different routing algorithms.

forwarded within our attacker model.

Finally, we observe that attacker placement remains an important consideration even when random routing is used. Applying the same strategy to placing attackers as in the deterministic case, we computed an average entropy metric of 0.61 for 100 attackers, as compared to an average of 0.83 for randomly placed attackers. This is particularly impressive as the standard deviation for randomly placed attackers is only 0.018.

5 Alternative Routing Algorithms

While the random routing algorithm presented in the previous section achieves better values of the anonymity metric than Chord routing, it does so at the cost of increased path length. The expected path length in Chord routing is $(\log_2 N)/2$, and the worst case is $O(\log_2 N)$. However, with random routing the expected path length increase to $(\log_2 N)^2$, with a worst case of $O(N)$. In this section, we consider some other choices routing algorithms and examine their path lengths and the anonymity they achieve.

One simple modification to improve the expected path length is to preferentially choose larger fingers when choosing a random finger for forwarding. For example, we can pick the best finger with probability $1/2$, the next best finger with probability $1/4$, and so on. The expected path length for this algorithm is only twice as long as in regular Chord routing, $\log_2 N$, though the worst case is still $O(N)$.

Kumar and Bansal [KB02] suggest another method of picking fingers for anonymous routing. At any node, they compute the distance between the destination and the current node d' (i.e. $d' = 1 - d$ where d is the distance between the node and the destination). Then they find the longest finger that makes progress towards the destination with the restriction that it is no longer than d'. The routing then picks between that finger and the next k smallest fingers at random. This method is motivated by the fact that nodes who are further away from the destination (large d) are more likely to be identified, so the smaller hops are taken to ensure better mixing of node paths in that case. The expected path length here is once again about that of Chord, for small values of k: for $d' \in [0, 0.5)$, the hops taken will look like a backwards Chord route, as larger and larger hops are taken each time. For $d' \in [0.5, 1)$, normal Chord routing is used. They call this method "random-finger-k."

Figure 6 plots the entropy metric computed for these routing algorithms. We created 10 networks of size 1000 and set a random 100 of them to be attackers, and ran a routing simulation for each of the routing algorithms on each network. We plot the measured entropy metric for the 10 runs in order from worst to best for ease of readability. As we expect, random routing outperforms all other algorithms; it also has the smallest variance. Weighted random performs nearly as well, which suggests that it may be a suitable compromise between performance and anonymity. Random-finger-2 produces significantly worse results, sometimes performing worse than unmodified Chord. We conjecture this is because the number of paths tried by this algorithm is still very restricted, thus the uncertainty of the attackers is not increased very much; however, the average paths are longer, and hence there is more of a chance of a message being intercepted by some attacker.

We also looked at the distribution of the probability of identification produced by each of these routing algorithms; the cumulative distribution is shown in Figure 7. As we expect, the randomized algorithms are more evenly distributed than the regular Chord routing, and the random-finger-2 algorithm most closely corresponds to the Chord distribution.

An important thing to note is that both of the new algorithms introduced in this section have a longer tail than random routing; this means that there is a larger population of nodes who are receiving low anonymity benefits. Unfortunately, whether a node belongs to this population is influenced by factors beyond most node's control: distance from the destination, indegree, and number of neighbors that are attackers. Further, the best indicator — proximity of attackers — is one that cannot be measured by the node. Hence a user cautious about his privacy will be reluctant to use these networks for fear of falling inside the poor anonymity section of the distribution. In the next section, we propose a modification to the Chord structure that attempts to overcome this problem.

6 Indirect Routing

As we saw above, even algorithms that produce high values of the entropy metric leave some nodes with a high probability of identification. A system that could offer some degree of anonymity to *all nodes* rather than just guarantee good anonymity *on average*, is appealing.[3]

The factors responsible for high probability of identification include a node's indegree, proximity of attackers, and the distance from the destination, all factors that are difficult to change. The last factor could be changed by altering the destination, but that would affect the semantics of the routing. Another approach would be to use a different destination as an intermediate point for delivering a message to the true destination. Choosing the intermediary at random would introduce another random factor, meaning that each nodes will have a high expected level of anonymity, though some messages will be less anonymous with low probability, rather than a few nodes obtaining a low level anonymity at all times.

6.1 Naive Method

Let us consider a naive method for routing through an intermediary. Assume node s wishes to look up address d. To do this through an intermediary, he picks a random ID $r \in [0, 1)$ and routes a request of the form (r, d), which is interpreted as "to r, please query d for me." When the node responsible for r, the intermediary, receives this request, it queries d normally and returns the result.

Ideally, we would like to take advantage of the fact that there are essentially two queries — from the originator to the intermediary and from the intermediary to the destination — and require the attacker to intercept both to link the originator and the destination. However, since the destination is included in the first query, an attacker on the first path can simply parse message to extract it. In particular, attackers close to the source still have a high probability of linking the source and destination together.

6.2 Split Routing

It is not necessary to reveal the destination to the nodes along the path to the intermediary in order for them to be able to forward the message, so we consider using cryptography to hide it from them. However to encrypt a message so that only r can read it, the originator needs to obtain some key K_r; doing this securely is difficult as each node only maintains local knowledge, and does not know the identity of r. Instead, we use a much simpler cryptographic tool that requires no key material, in a method we call *split routing*. The idea is to send off several *shares* of the routing request to the intermediary, such that:

1. the shares are built using an m-of-n secret-sharing scheme: m shares must be captured (of n sent) in order to reconstruct the the true destination, and any collection fewer than this number of shares reveals *no* information about the destination.

2. the shares take paths that are not likely to converge except at the intermediary.

If these conditions can be achieved, the attacker's task of learning the true destination is greatly complicated. To directly learn the true destination from the routing request, he must intercept the required number of shares on their way to the intermediary. Since we attempt to route along paths that are as disjoint as possible, the attacker has to get lucky and be positioned well on several paths.

We show in the following sections how a simple version of split routing (using 2-of-2 sharing) can be accomplished in Chord.

[3]Of course, a node has *only* attackers for neighbors is hopeless.

6.2.1 Message Splitting

A 2-of-2 split is simple to produce. Given the original routing request (r, d), we wish to split it into two messages, (r, c_0) and (r, c_1) such that with both, it is possible to determine d, but without both, nothing about d is learned.

The source simply picks a *mask* $m \in [0, 1)$ at random and sets $c_0 = m$ and $c_1 = m \oplus d$. Since $c_0 \oplus c_1 = d$, it is certainly possible for the intermediary to recover the true destination. On the other hand, both values c_0 and c_1 are independent of d, so nothing is learned about d by knowing just one or the other.

6.2.2 Bidirectional Routing

All that is left is for the source to route the two messages (r, c_0) and (r, c_1) to r along paths that are highly unlikely to converge — ideally, they should meet only at the node responsible for r.

There is an obvious (and somewhat arbitrary) asymmetry in Chord that suggests an extension with guaranteed disjoint paths. Standard Chord routes clockwise (CW), where it is assumed that the address space increases in the clockwise direction. If the source node has id s, his CW fingers point to nodes responsible for addresses $s + \frac{1}{2}, s + \frac{1}{4}, \ldots, s + \frac{1}{2^i}, \ldots$, where it is assumed that the addition "wraps around" back to $[0, 1)$.

We can augment Chord with counterclockwise (CCW) fingers. Again, if s is the ID of the source node, his CCW fingers point to those nodes responsible for the addresses $s - \frac{1}{2}, s - \frac{1}{4}, \ldots, s - \frac{1}{2^i}, \ldots$, where it is assumed that the subtraction operation "wraps around" back to $[0, 1)$. Figure 8 depicts some of the CW and CCW fingers of node s.

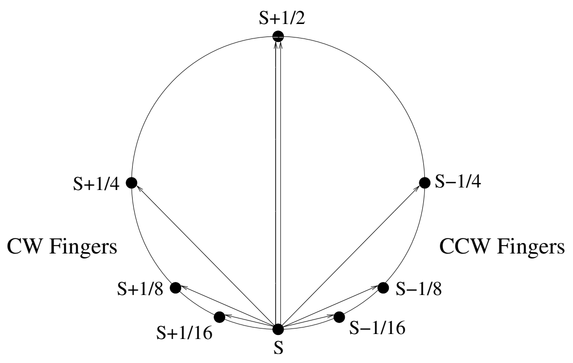


Figure 8: Bidirectional Chord.

With both CW and CCW fingers, it is possible to route messages along paths that do not converge until they get to the destination. Thus the source can send the routing request shares (r, c_0) and (r, c_1) to r along the CW and CCW fingers, respectively.

6.3 Analysis of Bidirectional Routing

To analyze split routing, we first identify the kinds of observations that are helpful to the attacker. We only consider the case when the attacker is able to recover the true destination, and treat other messages as background traffic, since the aim of the attackers is to identify who is communicating with the destination. In particular, if an attacker were to intercept only one share on its way to the intermediary, it would not be able to draw any inference since the destination remains hidden. Similarly, intercepting only the message from the intermediary to the destination is not helpful, since even if the intermediary is completely identified, its identity reveals no information about the source since it was picked at random.

Thus, the attacker only gains information if it can intercept the message along two of the three paths. If one of the paths is the path to the true destination, the attacker must still correlate the messages on the two paths, either by using timing analysis or by looking at those messages that are likely to have come from r (since the final message is routed recursively and hence does not directly reveal r's identity) and reaching a probabilistic conclusion. Both of these tasks are much harder to do if many messages are being routed through the network. We thus focus on the case that the two intercepted messages lie on the two disjoint paths between the originator and r.

In this case, the originator can immediately deduce the true destination (just as he could by intercepting a single message in standard routing), but he also gets two *corroborating* observable events indicating the true source. This is quite different from the standard routing case where an intercept is not necessarily so devastating. A split intercept will typically induce a very low entropy conditional distribution on the possible sources. This is especially true with larger numbers of shares — more shares means the source is less likely to be detected, but if it is detected, its identity is almost surely revealed. Therefore, split routing is really an all-or-nothing game, the rule is *don't get intercepted!*.

6.3.1 When Split Routing Wins

Since split routing squares[4] the probability of intercept, split routing really does best where there are relatively few attackers. For example, in a network saturated with attackers, suppose the probability of an intercept along a routing path is $p = .99$ (as a typical node in a network with. Split routing reduces this probability to $p^2 = (.99)^2 \approx .98$, which not really an improvement – and now an intercept means more likely identification, to boot! On the other hand, in a network with relatively few attackers, the intercept probability may be more like $p = .1$, and split routing will drive this down to $p^2 = .01$.

What is the benefit of split routing then if the probability of intercept is already low? Our experiments show that it, in addition to reducing overall entropy loss, it also improves the worst-case probability of identification for nodes that are in a poor position in a network. For these nodes, their individual level of anonymity with direct routing is much lower than average, and thus they are most affected by the reduced probability. They also benefit because the random choice of an intermediary will affect their probability of identification and they will only experience the worst-case values for a small fraction of the messages they send.

The final benefit of split routing is that it gives the participants some control over their anonymity. Users desiring higher anonymity can use split routing with a greater number of shares. Even though it will not possible to route them all along disjoint paths, the probability of an attacker intercepting all shares does decrease with a larger number of shares, assuming they is some random choice involved in what routes they take. (For example, a node can give a share to each of its neighbors and ask them to forward it to the intermediary.) A chained scheme with several intermediaries can also be used, though the details of splitting shares become more complicated.

7 Future Directions

7.1 Analyzing Entropy

While the entropy metric we used has a nice interpretation and captures an important aspect of anonymity (that is, non-uniform probability distributions), it can be difficult to apply to complex, dynamic systems. We use an empirical approach, described in Section 3.4, to compute the entropy metric. We use two methods to calculate the underlying probability distributions: sampling and non-deterministic simulation. The former can produce imprecise results, while the latter is quite expensive and does not scale well to large-sized networks. Research into other more efficient or precise methods of approximating the entropy can help expand the range of systems and scenarios that can be analyzed using our empirical methodology.

Furthermore, we need to better the understand how to combine the entropy computed for single observations to obtain results for a set of observations. Our empirical methodology is unable to consider sets of more than a few observations as the number of possible combinations grows exponentially. An analytical way of combining entropies would allow us to analyze more complex situations, as well as study the longitudinal anonymity of a system.

7.2 Other Metrics

As discussed in Section 3.3, entropy is a measurement that talks about *expectation*, and thus only describes the average anonymity characteristics. Indeed, in Section 4 we saw that the metric was unable to capture all of the behavior that was relevant to our analysis of anonymity, such as the existence of particular nodes that are relatively exposed despite a good overall entropy. An entropy metric that better corresponds to the real requirements people might place on anonymity systems would be helpful for performing more meaningful analyses.

7.3 Other Structured Peer-to-Peer Networks

We studied Chord since it is relatively simple to analyze, yet still exhibits a rich variety of interesting properties with respect to anonymity. Fortunately, many of these properties (e.g., irregular indegree, small outdegree) are shared with other structured peer-to-peer networks, so it should be possible to generalize our analyses to these networks. Most of our modifications rely on having a choice of paths to use in routing. Different routing geometries provide varying amounts of flexibility in path choice [GGG$^+$03]; our modifications would apply best to those networks that have a lot of choice. Analyzing other networks, more could be learned about not only the properties of the particular networks

[4]Here we assume 2-2 share split routing, but the discussion generalizes.

with respect to anonymity, but also perhaps anonymity properties common to all structured peer-to-peer networks. An interesting question is whether other structures are better suited for anonymity than Chord.

7.4 Unstructured Peer-to-Peer Networks

Although our privacy-enhancing modifications be harder to apply to unstructured networks such as Freenet [CSWH00] and GNUnet [BG03], it would certainly be worthwhile to apply our methods for calculating the entropy metric to these networks. A uniform measure of anonymity that is also easy to calculate on arbitrary networks would be enormously helpful for purposes of comparing all peer-to-peer networks.

7.5 Performance and Anonymity

It is commonly held that there is a trade-off between performance and anonymity; certainly, in our study, the algorithms that provided best anonymity came with an associated performance cost. An important question is how various performance optimizations, such as locality-aware routing, affect anonymity. Modeling locality information introduces a lot of complexity to the analysis; however, our empirical approach to measure entropy is a promising way to avoid this complexity through simulation. Understanding the trade-off between performance and anonymity will be vital to the design of future efficient anonymous systems.

8 Conclusion

We found that Chord, and probably most structured peer-to-peer networks in general, have particular properties that are quite relevant to the anonymity of participants in these networks. We identified some of these properties (indegree, proximity to attackers, proximity to destination, etc.) and proposed ways (such as randomization during routing) to ameliorate the negative effects while accentuating the good effects. What we learned will be useful not only in designing new peer-to-peer networks with an eye on anonymity, but also in employing a universal metric for objectively comparing existing systems.

9 Acknowledgments

We would like to thank Roger Dingledine, Jayanth Kumar Kannan, David Molnar, and Bryce Wilcox for many helpful discussions on the topic of anonymous systems. In addition, the anonymity bibliography that Roger Dingledine maintains[5] proved to be an invaluable tool in our research.

References

[BG03] Krista Bennett and Christian Grothoff. GAP – practical anonymous networking. In Roger Dingledine, editor, *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*. Springer-Verlag, LNCS 2760, March 2003.

[BGS01] Adam Back, Ian Goldberg, and Adam Shostack. Freedom systems 2.1 security issues and analysis. White paper, Zero Knowledge Systems, Inc., May 2001.

[Cha81] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 4(2), February 1981.

[Cle] The Cleaner. Gnutella wall of shame. http://www.zeropaid.com/busted/.

[CSWH00] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pages 46–66, July 2000.

[5] `http://freehaven.net/anonbib/`

[Dai96] Wei Dai. Pipenet 1.1. Usenet post, August 1996.

[Dan03] George Danezis. Mix-networks with restricted routes. In Roger Dingledine, editor, *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*. Springer-Verlag, LNCS 2760, March 2003.

[DS02] Roger Dingledine and Paul Syverson. Reliable MIX Cascade Networks through Reputation. In Matt Blaze, editor, *Proceedings of Financial Cryptography (FC '02)*. Springer-Verlag, LNCS 2357, March 2002.

[DSCP02] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*. Springer-Verlag, LNCS 2482, April 2002.

[FM02] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, DC, November 2002.

[GGG$^+$03] Krishna P. Gummadi, Ramakrishna Gummadi, Steven D. Gribble, Sylvia Ratnasamy, Scott Shenker, and Ion Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of ACM SIGCOMM 2003*, August 2003.

[HB] Ian Hall-Beyer et al. Gnutella. http://www.gnutella.com/.

[HW02] Steven Hazel and Brandown Wiley. Achord: A variant of the chord lookup service for use in censorhip resistant peer-to-peer publishing systems. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02)*, Cambridge, MA, March 2002.

[KB02] K. Jayanth Kumar and Matulya Bansal. Anonymity in chord. http://www.cs.berkeley.edu/˜kjk/chord-anon.ps, December 2002.

[RP02] Marc Rennhard and Bernhard Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2002)*, Washington, DC, USA, November 2002.

[RR98] Michael Reiter and Aviel Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.

[SD02] Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Roger Dingledine and Paul Syverson, editors, *Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, San Diego, CA, April 2002. Springer-Verlag, LNCS 2482.

[SM02] Emil Sit and Robert Morris. Security considerations for peer-to-peer distributed hash tables. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02)*, Cambridge, MA, March 2002.

[SMK$^+$01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM 2001*, August 2001.

[STRL00] Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an Analysis of Onion Routing Security. In H. Federrath, editor, *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pages 96–114. Springer-Verlag, LNCS 2009, July 2000.

[WALS02] Matthew Wright, Micah Adler, Brian Neil Levine, and Clay Shields. An analysis of the degradation of anonymous protocols. In *Proceedings of the Network and Distributed Security Symposium - NDSS '02*. IEEE, February 2002.

[WALS03] Matthew Wright, Micah Adler, Brian Neil Levine, and Clay Shields. Defending anonymous communication against passive logging attacks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003.