Defending Anonymous Communications Against Passive Logging Attacks *

Matthew Wright† Micah Adler† Brian Neil Levine† Clay Shields⋆

mwright@cs.umass.edu micah@cs.umass.edu brian@cs.umass.edu clay@cs.purdue.edu
† Dept. of Computer Science, University of Massachusetts, Amherst, MA 01003
⋆ Dept. of Computer Science, Georgetown University, Washington, DC 20057

Abstract

Using analysis, simulation, and experimentation, we examine the threat against anonymous communications posed by passive logging attacks. Previous work analyzed the success of such attacks under various assumptions. Here, we evaluate the effects of these assumptions more closely. First, we analyze the Onion Routing based model used in prior work in which a fixed set of nodes remains in the system indefinitely. We show that for this model, by removing the assumption of uniformly random selection of nodes for placement in path, initiators can greatly improve their anonymity. Second, we show by simulation that attack times are significantly lower in practice than bounds given by analytical results from prior work. Third, we analyze the effects of a dynamic membership model, in which nodes are allowed to join and leave the system; we show that all known defenses fail more quickly when the assumption of a static node set is relaxed. Finally, we address the question of whether the regular communication patterns required by the attacks exist in real traffic. We collected and analyzed the web requests of users to determine the extent to which basic patterns can be found. We show that, for our study, frequent and repeated communication to the same web site is common.

1 Introduction

Designing systems for anonymous communications is a complex and challenging task. Such systems must be secure against attackers at a single point in time; more importantly, they must also protect users from attacks that seek to gain information about users over the lifetime of the system.

In our prior work [17], we analyzed such an attack: the *predecessor attack*. In this attack, a set of nodes in the anonymous system work together to passively log possible initiators of a stream of communications. With sufficient path reformations — which are unavoidable in practice — the attackers will see the initiator more often than the other nodes. In that prior work, we showed that this attack applied to a class of protocols that included all protocols for anonymous communications that were known at the time. We also gave an analysis that placed bounds on how long the attacks would take to run for a number of specific protocols.

In constructing the attack and analysis in that paper, we made several simplifying assumptions about how the protocols operated. Here we examine the effects of relaxing each assumption. Specifically, we assumed:

1. The subset of nodes that forward an initiator's messages are chosen uniformly at random;

*This paper was supported in part by National Science Foundation awards ANI-0087482, ANI-0296194, and EIA-0080199.

2. Users make repeated connections to specific *responders*, which are outside points of communication;

3. Nodes do not join or leave the session;

These assumptions were necessary for the proof we provided that showed that the attack works in all cases, and they were also critical to our analysis of the bounds on the time required for a successful attack. We argued in that paper that the assumptions are reasonable and that the attack works in many scenarios where the assumptions are violated.

In this paper, we examine more closely the universal applicability of predecessor attacks against anonymous protocols. We examine our assumptions and the effect relaxing those assumptions has on the effectiveness of the attack. Our specific contributions are:

- First, we show that defenses that use non-random selection of nodes for path creation offer significant protection for the initiator.

- Second, we show that dynamic membership of nodes can have a significant detrimental effect on initiator anonymity. The influence of dynamic membership is determined by the actual distribution of node session lengths.

- Third, we examine the practical effectiveness of the attack through simulation. Our previous work proved analytical upper bounds on the number of rounds required; the simulations in this paper demonstrate that attackers can be successful in significantly fewer rounds than the maximums guaranteed by the bounds. E.g., attacks on Onion Routing and Crowds, with 1000 nodes and 100 attackers, succeed in a time one-fifth of the rounds guaranteed by the upper bounds.

- Fourth, we characterize measurements taken from a web proxy to show the actual frequency and duration of user activities on the Web. This study allowed us to make some observations about user behavior with regard to our assumptions.

The body of this paper is organized around those goals. In Section 2, we review related work. We then present and analyze the effectiveness of new techniques for avoiding predecessor attacks in Section 3. We describe the results of our simulations of the predecessor attack in Section 4. In Section 5, we show how often users go to the same website from data obtained by tracking real users. We offer concluding remarks in Section 6.

2 Background

As we seek to build on our previous work, we explain our results from that work in this section. We also describe related material.

2.1 Our Prior Work

Our previous work described the predecessor attack, first discovered by Reiter and Rubin as an attack against Crowds [13]. The primary purpose of our current work is to extend our previous results in the analysis of anonymous protocols. In this section, we review the definitions, methods, and results of that work, and we refer the reader to the full paper if greater detail is desired [17].

The first contribution of our previous work was to define a class of protocols, which included all known protocols for anonymous communication, and to prove that the class degrades against the predecessor attack. We defined an *active set* as the set of nodes used by the initiator of a communication to propagate its message through the network. This can be, for example, a path in Crowds or the set of nodes that share coin flips with the initiator in a DC-Net.

For any protocol inside our class, we required that the active set be chosen uniformly at random many times. In current protocols, that assumption holds because active sets change each time a node is allowed to join the network. If active sets did not change, then the members of the active set of a node that just joined the network would know that they are propagating messages of that node, thereby exposing its communications. However, it is not necessary that the new active sets are chosen uniformly at random — in Section 3, we explore ways to choose paths that exploit this fact, with the intention of defending against degradation of anonymity.

The second major result in our prior work was a set of analytic bounds describing how long it might take for attackers using the predecessor attack to effectively degrade the anonymity of a user in Crowds, Onion Routing, and Mix-Nets. We gave bounds on the number of *rounds*, i.e., periodic changes in the active set, that guarantee for the attackers a high probability of success in guessing the initiator. We use the following notation: n is the number of nodes in the network, c is the number of those nodes that are attackers, and l is the fixed path length of Onion Routing or Mix-Nets.

Against Crowds, the attackers require $8\frac{n}{c}\ln n$ rounds to identify the attacker with high probability $\frac{n-2}{n}$. For the same level of confidence, the attackers need $8\frac{n^2}{c^2}\ln n$ rounds against Onion Routing and $8\frac{n^l}{c^l}\ln n$ rounds against Mix-Nets. In Section 4, we provide simulation results that are tighter than these bounds and show how the confidence of the attackers grows over time.

In the prior work, we also assumed that rounds occurred regularly, and that the initiator communicated with the responder in every round. If nodes are allowed to leave and join the protocol, then attackers can force rounds to occur as often as the system allows by simply having corrupt nodes join and leave. However, they cannot force the initiator to communicate with the responder during a round, which is necessary for the attackers to get data on the identity of the initiator. If the initiator rarely communicates with the responder, then the amount of time it takes for the attackers to get data from enough rounds can be very large. In Section 5, we use logs of real web usage to examine how many rounds attackers can expect to get data from over time.

2.2 Related Work

A number of papers have addressed attacks against systems of anonymous communications. The creators of Crowds [13], Onion Routing[16], Hordes [10], Freedom [1], Tarzan[7], Stop-and-Go Mixes [8], and others have provided analysis of their protocols against some attacks.

Only a few of these analyses consider the degradation of anonymity over time, including Reiter and Rubin's seminal work [13]. Berthold, *et al.*, discuss an intersection attack against the anonymity groups that arise when multiple mix routes are chosen [3]. In this attack, the different anonymity groups are intersected with each other to shrink the number of possible initiators.

Raymond also discusses an intersection attack based on observations of user activity [11]. Only the active users can be the initiator, and the set of active users changes over time. Intersecting the sets of active users reduces the set of possible initiators. We explore this idea further in Section 3.3.1.

More recently, Shmatikov used formal analysis and model checking to verify the efficacy of the predecessor attack against Crowds [15]. Due to the high processing requirements of model checking, he was only able to demonstrate probabilities of attacker success for small numbers of nodes, i.e., twenty or less. In this paper, we use simulation to extend these results to thousands of nodes with an acceptable loss of precision.

One part of our work is to study users' web surfing behavior to determine whether users visit the same site repeatedly over time. Given the large number of studies on the World Wide Web and user behavior on the web, one might believe that this work would be done already. We are unaware of any studies that show longer-term user behavior at the level of detail that we require.

Work by Baryshnikov, *et al.*, tries to predict traffic patterns based on content, but from the point of view of servers handling load and not considering single users [2]. Other papers on traffic prediction, including work by Duchamp [6] and work by Davison [5], model users based on recent content. In Duchamp, only the last 50 requests are used, while the newer work by Davison only uses the last five HTML documents. In this work, we seek patterns over much longer periods.

2.3 The Predecessor Attack on Recent Protocols

Recently, several significant protocols for anonymous communications have been published. In this section, we discuss some of these protocols and their resistance to the predecessor attack.

The first of these P5, by Sherwood, *et al* [14]. This protocol is designed for anonymity between peers connecting to each other, rather than outside responders. It could, however, be adapted to outside communication by using destination peers as the final proxy to the rest of the Internet. P5 uses a tree-based broadcast protocol, where a user's anonymity is based on the sizes of the different broadcast groups in which she is in.

The authors assume that "users do not leave once they join" to prevent a decline in users' anonymity [14]. Without this assumption, anonymity groups would shrink, leading to degradation of anonymity within the groups. We expect that this assumption does not hold well in today's networks, in which nodes may frequently shut down. When anonymity groups become too small, users must recreate a new communication tree, including a new key. This expensive step keeps the protocol from being vulnerable to the predecessor attack when the number of attackers is less than the size of the user's anonymity group.

The second protocol, Tarzan, by Freedman, *et al.* [7], is a peer-to-peer system built at the network layer. From the perspective of our analyses, which assume a peer-to-peer setting, Tarzan may be considered a variant of Onion Routing, as it uses Onion Routing-style layered encryption.

Tarzan achieves a higher level of practical security in some scenarios by having nodes select relays according to random domain selection. This means that attackers cannot overload the network with malicious nodes from within the same domain, as initiating nodes will not select proxies from that domain with any greater frequency. Attackers can gain an advantage, however, when the number of honest domains represented in the Tarzan group is small. An attacker may be able to operate nodes from domains with no honest Tarzan participants. With a few such domains, attackers could make it likely to appear on an initiator's path despite only operating a few corrupt nodes.

Additionally, two mutual anonymity protocols have been presented recently [18, 9]. These protocols, like P5, are designed to hide the identities of both communication parties from each other and from third parties. Both of these protocols are, like Tarzan, variants of Onion Routing, but with anonymity for both the client and server. Since they are designed for file sharing, paths are not maintained. This makes them highly vulnerable to the predecessor attack as new paths create more opportunities for attackers to be on the path. However, repeated connections to the same responder may not be as common for file-sharing as in other applications, and in fact easier to avoid.

3 Paths Based on Non-Uniform Peer Selection

In this section, we examine two main assumptions of our previous analysis of anonymous protocols.

- First, nodes are selected by the initiator uniformly at random for forwarding messages in the protocol.
- Second, nodes do not leave the session nor do new nodes join.

We show the effects of non-uniform selection of nodes during path selection, and we show the effects of membership dynamics on initiator anonymity. First, we consider a *static* set of nodes and attackers using passive logging and review a number of different path creation strategies. Then we analyze a *dynamic* model that incorporates the property that nodes can leave and join the system. We find that membership dynamics of the system greatly affect initiator anonymity.

Both models are based on the Onion Routing protocol [12]. We study Onion Routing because it is provides an intermediate level of both security and performance of path-based protocols. It is significantly more secure than Crowds against the predecessor attack, as described in our previous work. Additionally, the attack and defense models we study requires that the initiator be able to control exactly which nodes appear on a path to a responder, which Onion Routing provides and Crowds does not.

Briefly stated, the predecessor attack for Onion Routing is to use two attackers. When the attackers occupy the last node on the path and one earlier node, they log the node before the first attacker. The intuitive reason that the predecessor attack is successful is that the initiator is observed in more rounds than other nodes.

3.1 Model

We assume a set of nodes N, where the size of N is fixed at n. We also assume a set of attackers $C \subset N$, consisting of $c < n$ attackers. The remaining $n - c$ nodes in N are not trusted, but do not share information with the set of attackers. All nodes follow protocol forwarding rules correctly.

In our *static* model, the membership of C and N does not change. This property is not true in the *dynamic* model that we consider in the second part of this section.

Each node in N communicates uniquely with one corresponding *responder*, a node that is not a member of N. The responder does not collaborate with other attackers but it tracks the identity of nodes with which it communicates, and it cannot be trusted. In each round, each node creates a path of proxies, consisting of other nodes in N, between itself and its corresponding responder. It sends at least one message in each round. For the rest of this section we only consider a single initiator I, whose communications with responder R are being tracked by the attackers.

In this model, attackers may only passively log information obtained while running the protocol in the normal fashion. They have no ability to eavesdrop on or corrupt any other nodes in N. Nor do they disrupt the protocol by modifying messages, failing to deliver messages, or performing other denial-of-service attacks.

Within this model, we consider defenses in which I modifies the probability with which nodes can appear in the path. First we will consider what I can do by only modifying its own probability of appearing in the path. Then we will consider *fixed-node* defenses, in which some nodes always appear in certain positions on the path.

For this model, no mix-like techniques are used. Specifically, we assume a cost-free attack exists that allows attackers to determine if they are on the same path; e.g., analysis of interarrival of packets.

Defense Technique	Prob. of Success	If successful, rounds req., expectation	If successful, rounds req., Bounded w.h.p.
Static node membership model:			
[17] O.R. model	1	$(\frac{n}{c})^2$	$O((\frac{n}{c})^2 \ln n)$
Fixed placement of nodes			
First	$\frac{c}{n}$	$\frac{n-1}{c-1}$	$O(\frac{n-1}{c-1} \ln n)$
Last	$\frac{c}{n}$	$\frac{n}{c}$	$O(\frac{n}{c} \ln n)$
First and Last	$\frac{c^2}{n^2}$	1	1

Table 1: The probability of success and the number of rounds for a successful predecessor attack against various defenses in the static membership model.

Note that the attackers may be able to force rounds to occur with short, regular intervals by simply leaving and rejoining the system. The system must start a new round to allow the returning node to resume communications. If the system delays the start of the next round for very long, it will be a great inconvenience to the user of the returning node.

3.2 Static Membership Model

We considered a set of defenses for the static membership model that contrast the assumptions of our previous work of uniformly random path selection. We considered the following cases: The initiator fixes placement of nodes: fixed first position; fixed last position; fixed first and last positions; nodes fixed in other positions.

These are a significant set of cases to study, but we do not argue they are a complete set. A summary of our results is presented in Table 1.

3.2.1 Fixed First Node Defense

The next defense we consider in the static model is that of fixing nodes in certain positions on the path. There are several variations of this technique that depend on the positions that the initiator chooses to fix: first position, last position, first and last positions, and middle positions. Fixing the first position is equivalent to the *setup attack* of our previous work [17]; thus, this subsection describes a generalization of that technique. For all scenarios, we assume that the nodes for all other positions are selected uniformly at random from all of N. Additionally, the fixed nodes also are selected uniformly from all of N at the creation of the first path.

The first approach to a fixed-node defense is to use one other node continually in the first position on the path. We call the node selected for the purpose the *helper node*, H. This defense protects the initiator from ever being tracked by the attackers, except if the node picked as H is unfortunately an attacker. If H is not an attacker, then when the attackers run the predecessor attack on messages to R, they will see H as the initiator instead of I.

We now consider what happens when H is an attacker. We note that the initiator is not immediately exposed. An attacker must appear at the end of the path to see the initiator's messages to the responder and determine jointly with H that the two attackers are on the same path. Only then will the initiator be exposed.

The last node is selected at random from N, excepting H, and there is a $\frac{c-1}{n-1}$ chance that the node selected is an attacker.[1] Thus, in an expected $\frac{n-1}{c-1}$ rounds, I will be exposed as the initiator.

Since the probability of H being an attacker is $\frac{c-1}{n-1}$, that serves as an upper limit on the probability that the initiator is ever exposed. Due to the changing last node, the probability of I's exposure grows towards that upper limit as it becomes more likely that the last node on the path has been an attacker. This compares favorably with choosing all nodes on the path uniformly at random every round, which results in a probability of exposure that grows, albeit more slowly, towards one. We demonstrate this by simulation in Section 4; see Figure 5

3.2.2 Fixed Last Position

A slight variation on this approach is to put H statically as the last proxy in the path. This approach also keeps node I from being exposed, as long as H can be trusted, since all of the communications to R are hidden from the attackers. If, however, H is an attacker, then all of the initiator's messages will be observed by the attackers.

As with the fixed first position, an attacker selected as H will require another attacker to be selected as well before the initiator can be exposed. In this case, it is the first node, which will be an attacker with probability $\frac{c}{n}$. In $8\frac{n}{c}\ln n$ rounds, the initiator is identified by attackers, given that H is an attacker, with high probability $\frac{n-2}{n}$.

Again, there is a limit of $\frac{c}{n}$ on the probability of the attackers being successful, based on the chance of selecting an attacker for H. So the probability of the attackers being successful again grows towards it's maximum value of $\frac{c}{n}$ and becomes very close to that bound in fewer than $8\frac{n}{c}\ln n rounds$. (See Figure 5.)

3.2.3 Fixed First and Last Position

The third variation is to keep both the first and last positions on the path fixed to the same nodes. The two positions should be set to different nodes in N because if they were the same, then a single node could expose I as the initiator. In this scenario, if either the first or the last node is not an attacker, then I is safe from attack. If, however, both nodes are attackers, then I is exposed in the first round.

The probability of I having been exposed, then, is only $\frac{c(c-1)}{n(n-1)}$ and stays constant for all rounds of the protocol. Since fixing only one node leads to an eventual $\frac{c}{n}$ probability of exposure, this makes fixing nodes in both positions the best known choice for the initiator in the static model, given sufficient rounds. (See Figure 5.)

We study the effect of node instability on this approach in Section 3.3.

3.2.4 Middle Node

The remaining variation of fixing nodes statically in the path is for the initiator to position one or more fixed nodes somewhere after the first proxy, but before the last one. These fixed nodes will serve to obscure the initiator's identity, as more than one node will appear to attackers using the predecessor attack significantly more often than expected for non-initiators. Unfortunately, this defense has many problems.

For example, with one helper node in the middle of the path, the attackers can use the predecessor attack, treating the helper node as the responder, to determine that the initiator's path always goes through the helper node. A simultaneous predecessor attack with the initiator treated the responder will show that

[1] Note that I should never select H for the last position, as it allows H to immediately expose I as the initiator.

Defense Technique	Rounds for high prob. $\frac{n-2}{n}$ of attacker success
Dynamic node membership model:	
Intersection attack:	
Exponential dist. session with parameter λ	$\frac{-1}{\lambda}\ln(1-(\frac{n-2}{n})^{1/n})$
Pareto dist. session with parameter a	$\left(\frac{1}{1-(\frac{n-2}{n})^{1/n}}\right)^{1/a}$
Fixed first and last:	
Constant session length ρ	$\rho\frac{\ln(1-\frac{n-2}{n})}{\ln(1-\frac{c(c-1)}{n(n-1)})}$

Table 2: Number of rounds for a successful predecessor attack for various defenses.

the helper is not the initiator. The attackers can then distinguish between the two nodes, and identify the initiator correctly. Many variations of this defense are possible, as are many corresponding counterattacks.

3.3 Dynamic Membership Model

In this section, we examine the effects on initiator anonymity of membership changes to the set of proxies. We use the same model assumptions as in the static case, and add that nodes leave the system with independent and identical distributions. We call the length of the time they are active in the system their *uptime*.

3.3.1 Intersection Attack

If an attacker can obtain a collection of disjoint sets of nodes that each contain the initiator, then simply intersecting those sets can greatly narrow the attacker's search. If the intersection leaves only one node, this technique can completely expose the initiator. We refer to this technique as the *intersection attack* [11]. In this section, we describe how an attacker can use this attack and give analyses on how long it will take.

In existing peer-to-peer anonymous protocols, each participating peer knows the other available peers' IP addresses. This is a fundamental need for proper operation, because in these protocols peers communicate directly. This list of peers, however, gives the attacker exactly what she needs to perform the intersection attack.

To conduct this attack, the attacker only needs to keep a single peer in the system to obtain these lists. For every round in which the initiator contacts the responder, the attacker can add the list of peers to her collection of sets of nodes. As nodes leave and join the system, the intersection of those lists becomes smaller. In each round in which the initiator does not communicate with the responder, the attacker does not take a list for intersection. This increases the total attack time by one round each time it occurs.

The attacker can, over time, use the intersection attack to uniquely identify the initiator. Alternatively, when the attacker has reduced the set of possible initiators to a few nodes, the attacker can use other techniques to distinguish the initiator from the other nodes in the intersection. For example, the attacker could eavesdrop on the communications of the remaining nodes to identify the initiator through packet timing.

This attack does not reveal the initiator in all situations, because the technique requires that the communication between the initiator and the responder be uniquely identifiable. In other words, the initiator must be the only node in the system communicating with the responder, or there must be some identifying information in their communications. Two initiators to the same responder can cause each other to be removed from the intersection, as one node communicates with the responder in a round while the other node is not in the system. While it is possible to extend the attack to pairs of nodes and to even larger groupings, it makes the attack more complicated and time consuming.

Another caveat is that the attacker's list must include all currently participating nodes for the intersection to be accurate. The lists do not need to be exact, but they must not omit any participating nodes. A reasonable implementation of most protocols does not require such a complete list. The attacker can address this issue in some protocols by keeping more nodes in the system. This can help to make the list of currently participating nodes more complete. Also, attackers would not want to eliminate nodes unless they are known to be unavailable.

An attacker that can overcome these difficulties can use the intersection attack to quickly reduce the list of possible initiators. The key to this is the behavior of users in how long they stay connected to the system. We will model this uptime with an exponential distribution and a Pareto distribution. Pareto distributions have heavy-tailed behavior that corresponds roughly with the uptime of nodes observed in peer-to-peer file-sharing systems [4]. A Pareto distribution corresponds to some nodes remaining active for very long periods, while many other nodes join the system for only a short time.

We do not consider nodes that leave and join the system in the same round (i.e. if they become available in the immediate next round) as having left at all. Our distributions are constructed such that such short-term failures are not considered events. Note that nodes newly joining do not affect the intersection attack, nor do nodes that intermittently re-join.

In the exponential model, we say that the average uptime for a user is $\frac{1}{\lambda}$. Therefore, λ can be thought of as the failure rate of the user's node. The probability that a given node has left after T rounds is given by the probability distribution function, $F(T) = 1 - e^{-\lambda T}$. We model the nodes as independent and identically distributed (i.i.d.) processes. Thus, the probability that all $n-1$ nodes other than the initiator have left after T rounds is $P(T) = \left(1 - e^{-\lambda T}\right)^{n-1}$. We can also find the number of rounds, T, required for the attacker's chance of exposing the initiator to reach a value p. This is given by $T = -\frac{1}{\lambda} ln\left(1 - p^{1/(n-1)}\right)$. Note that T is linearly dependent on the average uptime, $\frac{1}{\lambda}$.

As we see in Figure 1A, the probability of all nodes leaving rises quickly to one. We show curves for the exponential distribution when the average uptime is one and four weeks, or 168 and 672 rounds, respectively. The round length is one hour, though the clock time required for the attack does not change with the round length. Longer round lengths may mean longer average uptimes, though, as it is more likely that a node that disconnects will be able to reconnect before the next round starts. We show results for a system with 1,000 total nodes.

After ten weeks, with an average uptime of one week, there is a 0.956 probability that all nodes have left except the initiator. When the average uptime is four weeks, the time to reach the same 0.956 chance that all nodes have left is 40 weeks. This agrees with the linear relationship between the average uptime and T.

The Pareto distribution gives us a different picture for how quickly the intersection attack works. For this model, we say that the average uptime is given by $E[T] = \frac{a}{a-1}$, where parameter a can be set give the desired average time[2]. The probability that a nodes leaves before time T is $F(T) = 1 - \frac{1}{T^a}$. Again, the nodes can be considered as an i.i.d. processes, and the probability that all $n-1$ nodes leave by time T is given by $P(T) = \left(1 - \frac{1}{T^a}\right)^{n-1}$.

We can rearrange this formula to find the number of rounds the attacker requires to expose the initiator

[2]We chose $a = 167/168$ so that $E[T] = 168$ hours, i.e., 1 week.

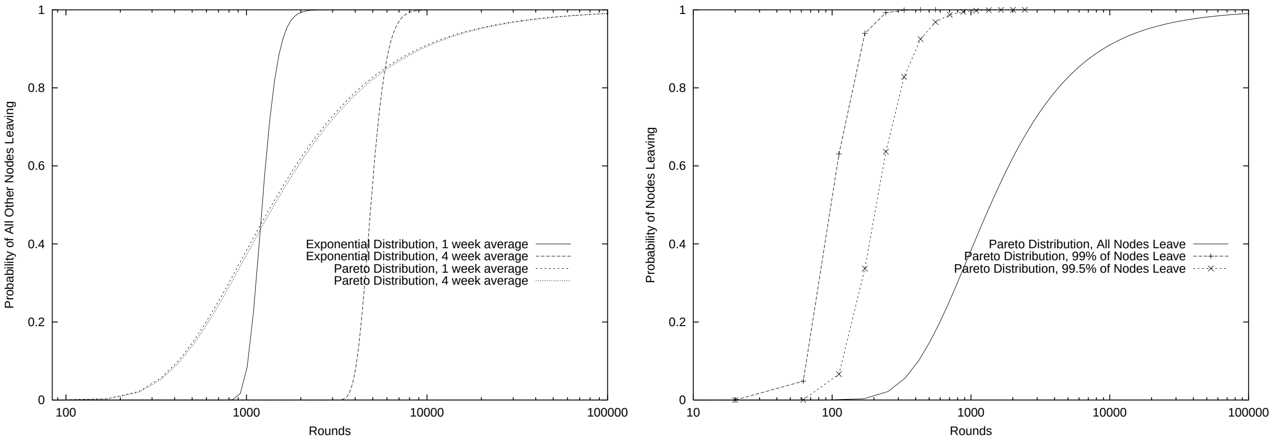


Figure 1: **Dynamic membership model:** The intersection attack. (A, left) The probability, for $n = 1,000$ nodes, of all nodes other than the initiator being intersected out of the set of possible initiators. (B, right) The probability of having less than ten and less than five nodes other than the initiator remaining in the system, and the probability of having no other nodes remaining, over time, where the original size is $n = 1,000$ nodes.

with probability $P(T) = p$: $T = \left(\frac{1}{1-p^{1/(n-1)}}\right)^{1/a}$. From Figure 1A, we see that the probability of total exposure, when all other nodes have left, grows much more slowly than in the exponential model. The chance of exposure becomes non-negligible in far fewer rounds, but the attacker is not guaranteed a high probability of success for a very long time. This is due to the heavy-tailed property of the Pareto distribution. Many nodes will leave the system early in the course of the attack, but it is likely that a few nodes will remain in the system for a long time. Also, the average time to leave makes little difference for the averages we used. We believe that this is because the chance of exposure mostly depends on whether other nodes remain for a long time, regardless of how much longer that time may be.

The possibility of long periods without complete exposure does not make the initiator safe from the intersection attack. This is because the number of possible initiators will quickly narrow to a small fraction of the remaining nodes. The probability that $k < n-1$ nodes will leave the system after T rounds is given by the binomial $P(T) = \Sigma_{i=k}^{n-1}\binom{n-1}{i}\sigma^i(1-\sigma)^{n-i-1}$, where $\sigma = \left(1 - \frac{1}{t^a}\right)$. In Figure 1B, we compare the probabilities that all but five and all but ten nodes leave the system, along with the probability that all nodes leave the system. We observe that the attacker reduces the list of possible initiators to a small fraction of the original nodes much faster than she can isolate the initiator completely. The attacker has a list of ten or fewer nodes with probability 0.999 in less than two weeks, when the average time for node failure is one week. In the same two weeks, the attacker can get a list of five or fewer nodes with probability 0.828.

3.3.2 Fixed Nodes with Instability

In Section 3.2.3, we described a method in which an initiator selects two nodes to permanently occupy the first and last positions on its path. This allows the initiator to keep attackers from linking it to any responder, as long as at least one of the selected nodes was not an attacker. This defense in the dynamic model is different as it depends on the stability of nodes: it requires the two chosen nodes to remain in the system as long as the initiator continues to contact a given responder. If the nodes leave, the initiator

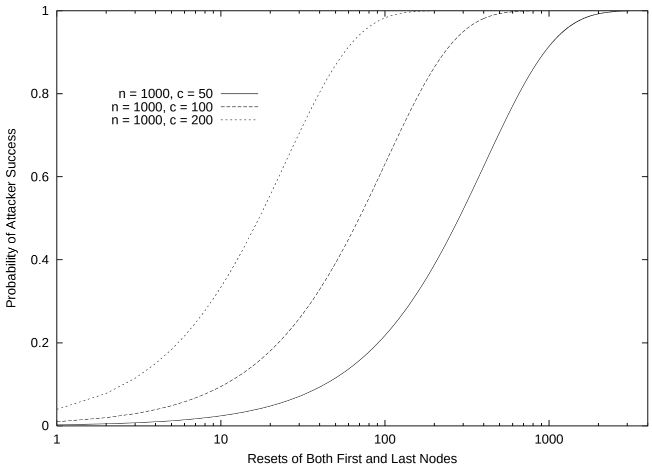


Figure 2: **Dynamic Model:** Probability of attackers' success when first and last nodes are fixed. The time is given in the number of resets needed. We show results for $n = 1,000$ nodes, and $c = 50$, $c = 100$, and $c = 200$ attackers.

must select new nodes to take their place. Every replacement of the nodes increases the chances that two attackers hold both positions simultaneously. The probability of exposure, then, grows closer to one over time.

We assume that nodes leave the network after a fixed, constant amount of time. In order to make comparisons with prior work and with results from Section 4, we measure time in rounds. Let us say that a node remains in the system for exactly ρ rounds. Also suppose that all nodes join the system at the same time. Then it is as if the system *resets* and forces new path selection every ρ rounds.

Every time the new first and last nodes must be selected, the initiator selects the new nodes uniformly at random from N. This creates a $P_1 = \frac{c(c-1)}{n(n-1)}$ probability of selecting attackers for both positions. We want to know the number of resets, R, such that the probability of attacker success, P_R, is at least some value p. Since $P_R = 1 - (1 - P_1)^R$, then substituting values of P_1, $R = \frac{\ln(1-P_R)}{\ln(1-\frac{c(c-1)}{n(n-1)})}$.

By using $P_R = p$, we get the number of resets, R_p, sufficient to make the attackers successful with probability p. If resets occur every ρ rounds, then $R_p\rho$ rounds are required for the attackers to be successful with probability p (See Table 2).

In Figure 2, we see that the probability of the attackers succeeding grows towards one as the number of resets increases. The similarity to simulation results for the predecessor attack in Section 4, for uniformly random path selection, suggests that fixing the first and last nodes may not provide significantly stronger security over time when nodes leave the system frequently. This is in contrast with the results shown for the static model in Figure 5, in which fixing the first and last nodes appears to be the best strategy. In general, fixing nodes on the path is only a good strategy when nodes leave the system only after long stays or not at all.

4 Simulating the Predecessor Attack

In our prior work, we provide upper bounds on the number of rounds required for attackers to perform the predecessor attack against nodes running Crowds, Onion Routing, and Mix-Nets. While these bounds helped to indicate the relative performance of each protocol against the attacks, we did not show them to be tight bounds.

In this section, we present simulation results showing the number of rounds required for attackers to have confidence that they have found the initiator. Additionally, we have simulated a number of the methods for selecting Onion Routing paths described in Section 3 and compare those results against uniform path selection.

4.1 Simulation Methodology

We simulated long-running Crowds-based and Onion Routing-based sessions. As we stated in the previous section, we believe these protocols offer the best tradeoffs of cost, performance, and security. In our simulations, one initiator sets up a path each round by selecting the nodes in the path. For each round, if attackers appear on the path, then they log the predecessor of the first attacker. Attackers in the Onion Routing model know if they are on the same path (e.g., by timing analysis in a real implementation).

Each data point of our graphed results are based on the average of ten *runs*. Each run consisted 10,000 separate simulations of path formation for a given numbers of nodes and collaborating attackers. Each run used a different seed for random number generation. After a specific number of rounds, the attackers for each of the 10,000 simulations made their guess of the initiator's identity based on their current logging information. We then determined the percentage of the 10,000 simulations that guessed correctly. This approach provides a confidence level for the attackers after any given number of rounds. Performing this ten times enabled us to give an average and determine variation in the results. In all, we ran over 66.8 million simulations. For all graphs we computed standard deviations, however they are not shown in most figures as they are too small to be significant and clutter the graph.

4.2 Results

Figure 3 compares simulation results of attacker success for Crowds and Onion Routing when the static membership model is used. The graph shows for both protocols the average number of simulations of 10,000 that correctly determined the initiator for a given number of rounds.

In all simulations, $n = 1,000$ and the average path length is 10. We show a dotted-line at $y = 5,000$ (50%) as that is the dividing line between *Probable Innocence*, where the attackers' best guesses are more than 50% likely to be wrong, and *Possible Innocence*, where the attackers best guesses are more than 50% likely to be correct [13].

We compare Onion Routing paths and Crowds paths of approximately equal length. It is not clear that the performance of each protocol with the same path length is equivalent, and Crowds path lengths will vary. If, however, network transfer times dominate the delay along paths for systems using either protocol, then equal path lengths provide a useful comparison. To make the path lengths similar, we selected several path lengths in Onion Routing and matched each of those with a Crowds' probability of forwarding (p_f) that gave the same average path length.

From Figure 3B, we see that Onion Routing lasts longer against the predecessor attack than a similar configuration of Crowds. The leftmost three lines at the 50% point are all for Crowds, while the three rightmost lines are for Onion Routing. For example, with $c = 100$, which gives us $\frac{n}{c} = 10$, the attackers require only 14 to 16 rounds against Crowds to get a 50% chance of exposing the initiator. For the same value of $c = 100$ in Onion Routing, the attackers require between 140 and 160 rounds. This ten-fold increase in the number of rounds appears to match the additional factor of $\frac{n}{c}$ found in the analytical results from our previous paper. This trend can be seen throughout the graph.

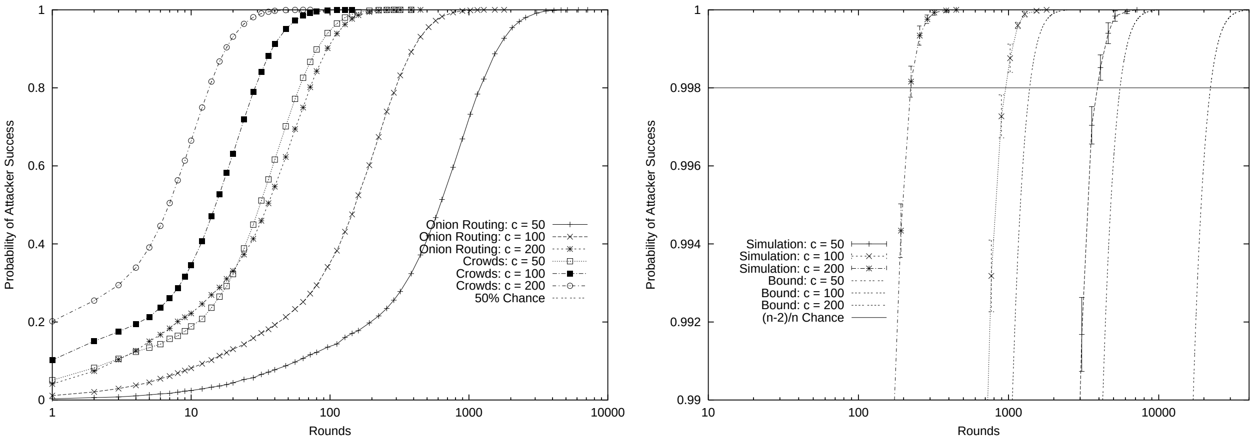


Figure 3: **Static membership model:** (A, left) Simulation of the predecessor attack against Crowds and Onion Routing, for varying numbers of attackers.(B, right) The predecessor attack against Onion Routing, top one percent.

We also see from Figure 3B that the larger the value of $\frac{n}{c}$, the longer the predecessor attack takes. For example, with Onion Routing, $c = 50$ attackers require between 576 and 640 rounds to find the initiator with 50% or better probability. This is about four times longer than for $c = 100$, for which $\frac{n}{c}$ is half as large. This is also in line with our analytical results for Onion Routing, which show that the number of rounds required has a squared dependence on $\frac{n}{c}$. This dependence holds for most values, but the differences are greater for low numbers of rounds. The linear dependence on $\frac{c}{n}$ for Crowds also holds for most values, again excepting the lowest counts.

In Figure 3A, we show the predecessor attack against Onion Routing as in Figure 3B, but only for when the probability of attackers guessing the initiator is greater than 0.99. A line is drawn where the attacker probability of success is $\frac{n-2}{n}$, which is the standard of high probability we set in the analysis from our previous paper and in Section 3

From the figure, we see that the number of rounds guaranteed by the bounds are much higher than the simulation results. The number of rounds required for the attackers to be logged with high probability $\frac{n-2}{n}$ is between 5.7 and 6.2 times less for the simulation than guaranteed by the bounds from our prior work.

We also see from the figure that the relationships between the numbers of rounds required for different values of $\frac{c}{n}$ holds as the attackers get closer to being certain that they have found the initiator. We use linear extrapolation between points to find how many rounds have passed when the lines cross the $\frac{n-2}{n}$ probability of attacker success line. With $c = 100$, approximately 4.3 times as many rounds are required than with $c = 200$ to reach the line. Between $c = 50$ and $c = 100$, the ratio is 4.1. The ratio predicted by the bounds is exactly four for each of these relationships.

We present simulation results for Onion Routing in Figure 4 with a fixed ratio of $\frac{n}{c} = 0.1$ and a fixed path length of $l = 10$, but with three values for n: $n = 100$, $n = 1,000$, and $n = 10,000$. Also, we show the bound for $\frac{n}{c} = 0.1$, calculated from the formula of our previous paper.

As in Figure 3A, we see the significant difference between the predicted values of the bounds and the simulation results. We also see the slight, but significant, difference in the upper half of the graph between the line for $n = 100$ and the lines for $n = 1,000$ or $n = 10,000$. This difference is not shown in the line given by the bounds and is not reflected in the insignificant difference in the results between $n = 1,000$ and $n = 10,000$.

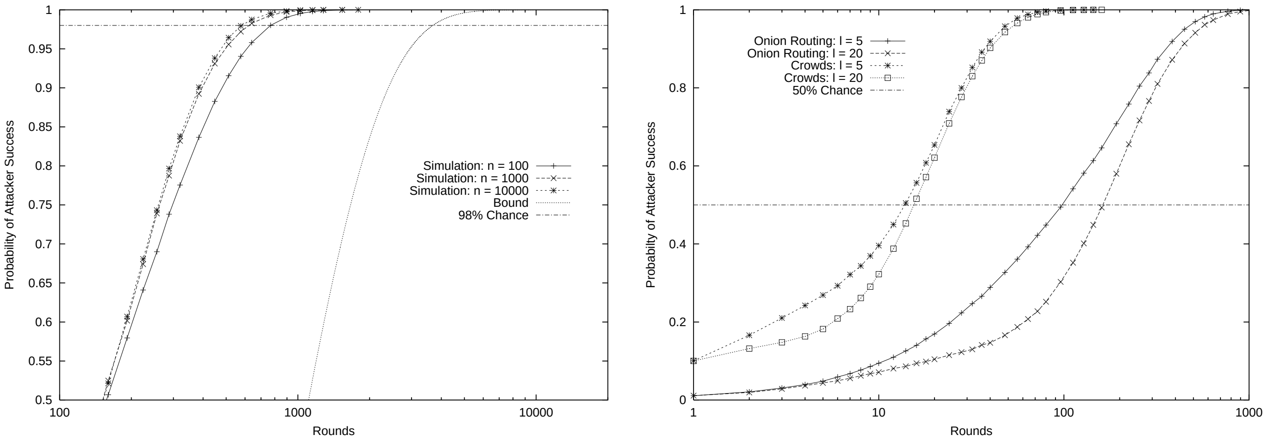


Figure 4: **Static Model:** (A, left) The predecessor attack against Onion Routing, for varying n. (B, right) The predecessor attack against Onion Routing and Crowds, for varying average path lengths

When $n = 100$, there are significantly fewer "non-initiators" than when $n = 1,000$ or more. The attackers are logging which nodes are logged on the initiator's path. Nodes are selected uniformly at random to be on the path. If there are fewer nodes from which to select nodes on the path, it is more likely that any one non-initiator will be selected with a high frequency and logged by attackers multiple times. Thus, when there are fewer nodes to select from, other nodes may be logged as often as the initiator for more rounds than when there are many nodes to select from.

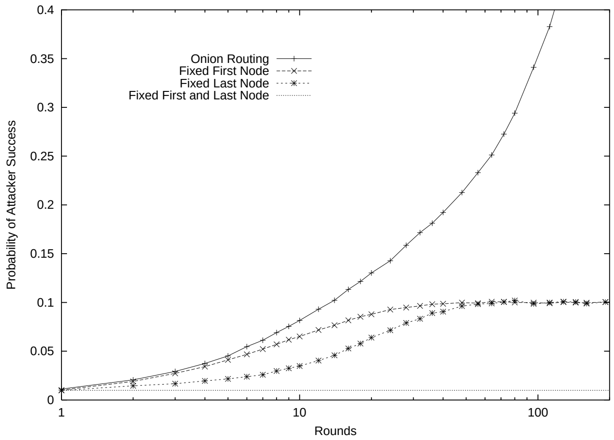
This effect can be seen in the bounds from our prior paper, although it does not effect the curves shown in our graphs. This is because the standard for high probability of attacker success, $\frac{n-2}{n}$, has an inverse dependence on n. The number of rounds required to reach a high probability that the attackers correctly identify the initiator is not dependent on n because the value of the probability changes with n.

The other significant parameter in these simulations is the average path length. In Onion Routing, this path length is fixed, while in Crowds, the path length depends on the probability of forwarding. We compare the two protocols with the same average path length in Figure 4.

From the figure, we see that Onion Routing significantly outperforms Crowds for all path lengths. We also see that longer path lengths outperform shorter path lengths. For example, after 96 rounds, attackers against Onion Routing with path lengths set to five have a 49.4% chance of identifying the initiator. With path lengths set to 20, however, the attackers only have a 30.3% chance of success.

Also in Figure 4, we see that average path length is more significant before the probability of attacker success has reached 50% than afterwards. Also, the difference between different path lengths is larger for Onion Routing, while largely insignificant for Crowds.

The success of fixing nodes in the path in the static model can be seen in Figure 5. After only ten rounds, the fixed first and last strategy is already a better defense than allowing these nodes to vary with each round. All three fixed strategies stay at a low probability of attacker success, while attackers see significant gains in the probability of their success against ordinary Onion Routing.



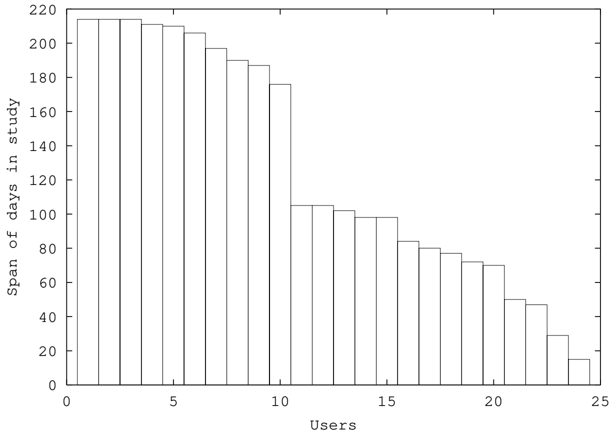


Figure 5: **Static Model:** Predecessor attack against Onion Routing when paths are selected uniformly at random, and when the first and/or last node is fixed. We show results for $n = 1,000$ nodes, $c = 100$ attackers, and path length $l = 10$.

Figure 6: Length of time each user employed the web proxy.

5 A Study of User Behavior

For the attacks we have studied to be successful, they require that initiators communicate with responders over long periods of time. To determine whether real user exhibit such behavior, we obtained traces of communications from a web proxy in the Department of Computer Science at the University of Massachusetts, Amherst[3].

We analyzed two important results from the data. First, we wanted to know the length of time that users contact the same web site (i.e., the responder). It is critical to note that the predecessor attack works in rounds of a specific length. For example, when rounds last one day it does not matter how many times the user contacts the same site during that day. Shorter round lengths allow new users to join more frequently, but increase the vulnerability of existing users.

Second, sites that are contacted most frequently (i.e., in the most number of rounds) are also the most vulnerable. Therefore, we wanted to know what percentage of a users' web traffic was vulnerable to passive logging attacks.

We tracked 24 volunteer users for 214 days. The span of time each user used the web proxy is shown in Figure 6. Each user could set their browsers to use or not use the proxy at will, so some browsing activity was lost if the users wished.

Figure 7A tracks whether users in the study re-visted the same responder over time. If we set the round length to 15 minutes long, the 214-day study divides into 20,549 rounds. Of 7,418 initiator-responder pairs, 103 were seen in 80 or more rounds; 45 pairs in 160 or more rounds; and so on. Values for longer round lengths are also shown in Figure 7A. In other words, only a small percentage of connections are long-lasting. However, Figure 7B shows what percentage of all traffic is represented by long lasting sessions. The longest-lasting sessions make up approximately 10% of all traffic, representing a significant fraction of users' behavior.

In our simulation results from Section 4, we found that in an Onion Routing system with 1,000 nodes

[3]This tracking is an on-going experiment for which we continually endeavor to attract more users.

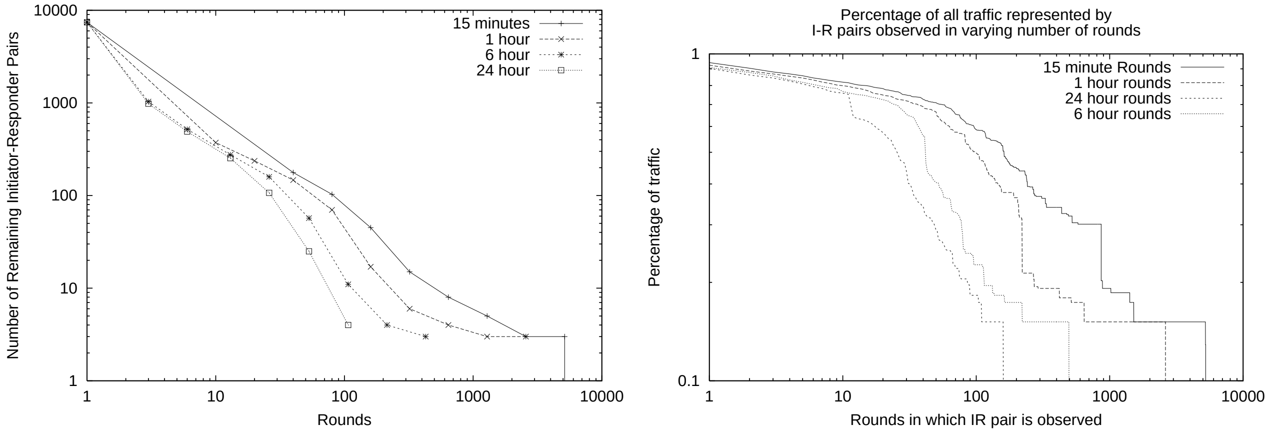


Figure 7: (A, left) Plot tracking if users contact the same web site over a long period of time. (B, right) Plot tracking if users contact the same web site over a long period of time.

and path lengths of 10, a set of 100 attackers required approximately 960 rounds to identify the initiator with an 99.8% probability of being correct. This matches the results for the five connections in our study seen in more than 1,284 fifteen-minute rounds. Thus, these five connections could have been tracked by attackers with a high probability of determining the initiators. Those connections make up 19% of all traffic, so a significant portion of the users' privacy would be compromised by such an attack.

If we relax the probability that the attackers are correct to 80%, we see that 15 different initiator-responder pairs can be tracked in the same time frame. These connections make up 36% of all the users' traffic. Over one-third of the traffic that went through the web proxy faced an 80% chance of linkage to the initiator.

6 Conclusion

In our prior work, we presented the first full analysis of the predecessor attack, an attack which seeks to identify the initiator of an anonymous connection over time. This work was important because it showed that all known anonymous protocols were subject to this attack; in some cases no attack against the protocol had been known.

In this paper, we have demonstrated several things. We have shown that the churn in group membership of anonymous protocols provides additional leverage for attackers to degrade the anonymity of initiators who stay in the anoymous group. We have shown by simulation that the upper bounds obtained in our prior work were truly upper bounds and that as a practical matter the attack can succeed much more quickly than previously described. We have also shown, through study of real network traffic, that users do seem to follow the necessary communication patterns for the attack to be successful over time.

These results are important for both the designers and users of anonymous protocols. It appears that designing a long-lived anonymous protocol is very difficult, and that users of current protocols need to be cautious in how often and how long they attempt to communicate anonymously.

References

[1] Adam Back, Ian Goldberg, and Adam Shostack. Freedom 2.0 security issues and analysis. Zero-Knowledge Systems, Inc. white paper, November 2000.

[2] Y. Baryshnikov, E. Coffman, D. Rubenstein, and B. Yimwadsana. Traffic Prediction on the Internet. Technical Report EE200514-1, Computer Networking Research Center, Columbia University, May 2002.

[3] O. Berthold, H. Federrath, and M. Kohntopp. Project anonymity and unobservability in the internet. In *Computers Freedom and Privacy Conference 2000 (CFP 2000) Workshop on Freedom and Privacy by Design*, April 2000.

[4] Jacky Chu, Kevin Labonte, and Brian Neil Levine. Availability and locality measurements of peer-to-peer file systems. In *Proc. ITCom: Scalability and Traffic Control in IP Networks II*, volume 4868, July 2002.

[5] Brian D. Davison. Predicting web actions from html content. In *The Thirteenth ACM Conference on Hypertext and Hypermedia (HT '02)*, pages 159–168, June 2002.

[6] Dan Duchamp. Prefetching hyperlinks. In *Second USENIX Symp. on Internet Technologies and Systems*, pages 127–138, October 1999.

[7] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *in Proc. ACM Conference on Computer and Communications Security (CCS 2002)*, November 2002.

[8] D. Kesdogan, J. Egner, and R. Buschkes. Stop-and-go-mixes providing probablilistic anonymity in an open system. In *Information Hiding*, April 1998.

[9] H. T. Kung, Scott Bradner, and K-S Tan. An ip-layer anonymizing infrastructure. In *Proc. MILCOM: Military Communications Conference*, October 2002.

[10] B.N. Levine and C. Shields. Hordes: A Protocol for Anonymous Communication Over the Internet. *ACM Journal of Computer Security*, 10(3):213–240, 2002.

[11] Jean-François Raymond. Traffic analysis: Protocols, attacks, design issues and open problems. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Proceedings of International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 10–29. Springer-Verlag, 2001.

[12] M. Reed, P. Syverson, and D. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communication Special Issue on Copyright and Privacy Protection*, 1998.

[13] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, November 1998.

[14] Rob Sherwood, Bobby Bhattacharjee, and Aravind Srinivasan. P5: A protocol for scalable anonymous communication. In *Proc. 2002 IEEE Symposium on Security and Privacy*, May 2002.

[15] Vitaly Shmatikov. Probabilistic analysis of anonymity. In *IEEE Computer Security Foundations Workshop (CSFW)*, pages 119–128, 2002.

[16] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an analysis of onion routing security. In *Workshop on Design Issues in Anonymity and Unobservability*, July 2000.

[17] M. Wright, M. Adler, B. Levine, and C. Shields. An analysis of the degradation of anonymous protocols. In *ISOC Symposium on Network and Distributed System Security*, February 2002.

[18] Li Xiao, Zhichen Xu, and Xiaodong Zhang. Low-cost and reliable mutual anonymity protocols in peer-to-peer networks. Technical Report HPL-2001-204, Hewlett Packard Laboratories, August 2001.