Passive-Logging Attacks Against Anonymous Communications Systems

MATTHEW K. WRIGHT
University of Texas at Arlington
MICAH ADLER and BRIAN NEIL LEVINE
University of Massachusetts Amherst
and
CLAY SHIELDS
Georgetown University

Using analysis, simulation, and experimentation, we examine the threat against anonymous communications posed by passive-logging attacks. In previous work, we 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 the 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. Fourth, intersection attacks against peer-to-peer systems are shown to be an additional danger, either on their own or in conjunction with the predecessor attack. 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.

This work is supported in part by grants from the National Science Foundation under awards EIA-0080119, ANI-0087482, ANI-0296194, and CNS-0549998 and in part by grant 2000-DT-CX-K001 from the U.S. Dept. of Justice, Office of Justice Programs. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the NSF or the DoJ.
Authors' addresses: M.K. Wright, University of Texas at Arlington, Dept. of Computer Science and Engineering, Arlington, TX 76019, USA; email: mwright@cse.uta.edu; M. Adler, University of Massachusetts, Dept. of Computer Science, Amherst, MA 01003, USA; email: micah@cs.umass.edu; B.N. Levine, University of Massachusetts, Dept. of Computer Science, Amherst, MA 01003, USA; email: brian@cs.umass.edu; C. Shields, Georgetown University, Dept. of Computer Science, Washington, DC 20057, USA; email: clay@cs.georgetown.edu.
This article is an extended version of Wright, Adler, Levine, and Shields [2003].

Categories and Subject Descriptors: C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed Applications*

General Terms: Security

Additional Key Words and Phrases: privacy, anonymity, anonymous communication, predecessor attack, intersection attack

ACM Reference Format:
Wright, M. K., Adler, M., Levine, B. N., and Shields, C. 2008. Passive-logging attacks against anonymous communications systems. ACM Trans. Inf. Syst. Secur. 11, 2, Article 7 (May 2008), 34 pages. DOI = 10.1145/1330332.1330335. http://doi.acm.org/10.1145/1330332.1330335.

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 [Wright et al. 2004], we analyzed such an attack: the *predecessor attack*. In this attack, the attacker controls a subset of the nodes in the anonymous system to passively log possible initiators of a stream of communications. With sufficient path reformations—which are unavoidable in practice—the attacker sees the initiator more often than the other nodes. In that prior work, we showed that the predecessor attack applies to a class of protocols that includes all existing protocols for anonymous communications. We also gave analysis that placed bounds on how long the attack takes to run for several specific protocols.

In constructing the attack and analysis in that paper, we made several simplifying assumptions about how the protocols operate. 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;
(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 these 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-uniform 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. We show how the defenses that use non-uniform selection of nodes fail in highly dynamic environments. We also present new analysis and simulation on a related passive-logging attack, the intersection attack, as applied to peer-to-peer settings. This dangerous attack leaves us with unresolved questions about how to design peer-to-peer systems for anonymous communications. We use more than one year of traces of Tor proxies collected by others to empirically evaluate the intersection attack.

—Third, we examine the practical effectiveness of the attack through simulation. Our previous work gave analytic upper bounds on the number of rounds required. The simulations in this paper demonstrate that, although the bounds provide a useful analytic tool to compare protocols, they are loose. The attackers can be successful in significantly fewer rounds than the maximums guaranteed by the bounds. For example, attacks on Onion Routing and Crowds, with 1,000 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.

There are two overall lessons from this article. The first is that the predecessor attack and the related intersection attack are both dangerous and realistic in proposed systems for anonymous communications. Users tend to behave in ways that can be tracked over time, leaving them subject to attacks that work more quickly than we previously realized. Dynamic membership in the system makes the environment even more dangerous for an initiator.

The second lesson is that the threat can be combatted. When system membership is not highly dynamic, we see that the fixed-node defenses from Section 3 are effective. Also, users can be warned to avoid extended communication patterns. Finally, we hope that the results presented here may shed light on possible defenses that might work more effectively against these attacks.

The body of this article 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. In Section 4, we study the predecessor and intersection attacks in dynamic environments. We describe the results of our simulations of the predecessor attack in Section 5. In Section 6, we show how often users go to the same Web site from data obtained by tracking real users. We offer concluding remarks in Section 7.

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 [Reiter and Rubin 1998]. 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 article if greater detail is desired [Wright et al. 2004].

The first contribution of our previous work was to define a class of protocols, which includes all known protocols for anonymous communication, and to prove that the anonymity of any protocol in 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 in our class, we required that the active set be chosen uniformly at random many times. In current protocols for open systems, 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. In the most expensive configuration of DC-Nets, the active set is the entire network and therefore does not change. However, a mostly static network, as assumed by protocols like DC-Net [Chaum 1988] and P5 [Sherwood et al. 2005], is not reasonable for open systems. Closed private networks, which could be static and offer other security benefits, face smaller anonymity sets due to restricted membership. In any case, we continue to limit our attention in this paper to open systems, which are the dominant architectures for anonymous communications today.

Another critical assumption of our prior work, as well as of this paper, is that the communication of users must be linkable from one session to another. Note that this will generally be the case for applications such as ssh or any Web service that uses a login or long-term cookies. *Hidden services*, which provides anonymity for servers, is an application that has received greater attention lately due to its implementation in Tor. Such a service is linkable across sessions due to the requirement that it be addressable. For other applications, such as email or general-purpose Web browsing with blocked cookies, the user's communications will not be linked as easily. However, the communications can be linked by profiling the user's behavior over time. Furthermore, any activity that occurs in the same session as a linkable activity is then linkable to other sessions. The accuracy of the attack would then depend on the fidelity of the profile generation and matching processes.

The second major result in our prior work was a set of analytic upper bounds on how long it takes 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 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 are assured of identifying the initiator in $8\frac{n}{c}\ln n$ rounds 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. These bounds were derived using Chernoff bounds, which do not provide a tight approximation. Additionally, we used a more easily analyzed algorithm that helped us give upper bounds on attacker performance, while the attacker can be more efficient in reality. In Section 5, 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 occur regularly, and that the initiator communicates with the responder in every round. The attacker can force rounds to occur as often as the system allows by simply having nodes leave and rejoin the system repeatedly. If, however, 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 6, we use logs of real Web usage as an example of how many rounds attackers can expect to get data from over time.

2.2 Related Analyses of Anonymity Systems

A number of papers address attacks against systems of anonymous communications. The creators of Crowds [Reiter and Rubin 1998], Onion Routing [Syverson et al. 2000], Hordes [Levine and Shields 2002], Freedom [Back et al. 2000], Tarzan [Freedman and Morris 2002], Stop-and-Go Mixes [Kesdogan et al. 1998], and others provide analysis of their protocols against some attacks.

Only a few of these analyses consider the degradation of anonymity over time, including Reiter and Rubin [1998] seminal work. Berthold et al. [2000] discuss an intersection attack against the anonymity groups that arise when multiple mix routes are chosen. In this attack, the different anonymity groups are intersected with each other to shrink the number of possible initiators. Raymond [2001] also discusses an intersection attack based on observations of user activity. 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 4.1.

A more sophisticated version of the intersection attack is the statistical disclosure attack. The original disclosure attack, as described by Kesdogan et al. [2002] takes sets of recipients from different times and uses intersection to find the recipients of a given user's messages. As this attack is NP-Complete, a statistical version is proposed by Danezis [2003] and extended by Mathewson and Dingledine [2004]. In the statistical version, the sets of recipients are divided into groups according to when the user is and is not sending messages. The set of recipients for that user is determined according to the relative frequency

with which each recipient receives messages when that user does and does not send.

The disclosure attack can be seen as orthogonal to the predecessor and intersection attacks studied in this work. The disclosure attack attempts to find the recipients for a given user. The predecessor and intersection attacks we study both attempt to find the sender associated with a set of observed messages or recipients. The two attacks can be used in the opposite way, e.g., applying the disclosure attack to every user to see who sent to a specific receiver. However, the attacks are not efficient when used in this way.

Berthold and Langos [2002] describe a method of defending against intersection attacks by using dummy traffic. It is not clear that their solution works in all systems, but it has the potential to limit the success of attacks such as those described in this article.

Shmatikov [2002] uses formal analysis and model checking to verify the efficacy of the predecessor attack against Crowds. 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., 20 or fewer. In this paper, we use simulation to extend these results to thousands of nodes with an acceptable loss of precision.

The Sybil attack, which applies to all open peer-to-peer systems, demonstrates that an attacker has the ability to control a significant fraction of the network [Douceur 2002]. Thus, the attacks we describe are of greater concern in such systems. Both the Tarzan and Morphmix peer-to-peer anonymous communications systems, however, implement a partial defense [Freedman and Morris 2002; Rennhard and Plattner 2004]. If we assume that an attacker will more easily control nodes within a single subnet than nodes evenly distributed throughout the Internet address space, their defense of first selecting subsets at random and then selecting nodes at random within the subset makes Sybil attacks less useful. However, recent developments in the use of *botnets*, networks of machines controlled by a hacker with the use of a trojan program, suggest that the machines would not be limited to a few subnets [Bächer et al. 2005; Honeynet Project 2005]. Margolin and Levine [2007] propose an economic approach for detection of Sybil attacks that can be applied to Tor and similar systems.

Other attacks on practical anonymity systems like Tor put these systems at substantial risk without the attacker needing the resources required for the predecessor attack. Murdoch and Danezis [2005] discovered a way to use the bandwidth limits of nodes to perform path traceback using only clients, i.e., without using any malicious Tor nodes. This attack may not scale well for large systems, as it was tested when Tor had only 35 nodes and requires full bandwidth utilization of a large fraction of the nodes during the lifetime of the path. Furthermore, Murdoch and Danezis [2005] propose a simple defense, making the bandwidth utilization of each path independent of other paths, which would stop their attack. Bauer et al. [2007] demonstrate that, when nodes select Tor nodes based on performance, attacks like the predecessor attack become much easier. This makes a strong case for not selecting

nodes based on performance, meaning that our more general approach is still important to understand.

Hidden services are an important application and are highly vulnerable to the predecessor attack (see related work [Goldberg and Wagner 1998; Scarlatta et al. 2001]). Øverlier and Syverson [2006] develop an attack that uses a combination of positional analysis, timing analysis, and the predecessor attack to significantly degrade the anonymity of users. Murdoch [2006] uses the clock skew induced by heat from loading a service's processor to identify a server with its service. This attack is rather challenging to stop, but a simple defense would be to make the service run at full CPU load all the time to keep the system temperature independent from the work load. Another simple defense would be to add noise to the time stamps used in TCP. Although it places a burden on the operator of the hidden service, these steps are prerequisites to maintaining security against the more powerful attackers we consider in this work.

Several past works study the efficacy of determining the source of encrypted traffic by comparison to known profiles. Sun et al. [2002] examine traffic analysis vulnerabilities that are specific to SSL Web traffic with high accuracy. Hintz [2002] provides a preliminary study of IPSec/SSH tunnel traffic analysis, and Bissias et al. [2005] and Liberatore and Levine [2006] provide more extensive empirical studies. Using only the packet sizes, the latter study is able to demonstrate an attacker accuracy of about 80 percent when identifying encrypted traffic sent by one of 2,000 profiled Web sites.

Independent works by Díaz et al. [2002] and by Serjantov and Danezis [2002] provide a strong case for new information theoretic metrics in anonymous communications. For our work, we have chosen to stay with a simple probabilistic notion that describes the chance that the attacker is successful in identifying the initiator. This facilitates comparisons with the analysis from our prior work as well as other prior works.

Finally, we note that the Tor system has implemented an extended version of our ideas from Section 3 on non-uniform path selection, in the context of protecting hidden services [Dingledine and Mathewson 2007]. Since the server provides a service that is highly linkable between sessions and may remain operating for some time, it could be quite vulnerable to the predecessor attack. Øverlier and Syverson [2006] discuss extensions to this idea, including *backup nodes* that provide greater reliability, and layered nodes which may provide stronger security. Scarlatta et al. [2001] suggest moving services from one peer to another as a way around this problem.

2.3 Web Traffic Measurement

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.

A number of extensive Web usage surveys have been conducted at the Georgia Institute of Technology [The Graphic, Visualization, and Usability Center 1998]. The surveys indicate that a significant fraction of users keep few bookmarks, which may indicate regular browsing patterns to relatively few sites. In the tenth survey, 38.8% of users reported keeping 50 or fewer bookmarks. Although this is more common for inexperienced users, 30.5% of users with four or more years of experience kept less than 50 bookmarks.

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 [Baryshnikov et al. 2005]. Other articles on traffic prediction, including work by Duchamp [1999] and Davison [2002], 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.

3. PATH SELECTION IN STATIC SYSTEMS

In this section, we remove the assumption from our prior work that nodes are selected by the initiator uniformly at random for forwarding messages in the protocol. In other words, we show the effects of non-uniform selection of nodes during path selection. We consider a *static* set of nodes and a subset of attacker-controlled nodes and review two different path creation strategies.

Our model is based on the Onion Routing protocol [Goldschlag et al. 1996]. We study Onion Routing because it provides an intermediate level of both security and performance among path-based protocols. It is significantly more secure than Crowds against the predecessor attack, as described in our prior work [Wright et al. 2002]. Fully connected DC-Net and Mix-Nets both provide higher security than Onion Routing, but they have significant additional costs, both for initiators and for intermediate nodes. Further, variations of Onion Routing have been implemented as systems for widespread use in Web browsing and other interactive applications [Dingledine et al. 2004; Back et al. 2000].

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 attacker-controlled nodes $C \subset N$, consisting of $c < n$ nodes that we call *attackers*. The remaining $n - c$ nodes in N are not trusted, but do not share information with the attacker. All nodes follow the protocol's 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 evaluate in Section 4.

Each node in N communicates uniquely with one corresponding *responder*, a node that is not a member of N. This assumption helps to simplify the analysis, but it may be possible to perform similar attacks based on a *profile* of responders built for each user [Mathewson and Dingledine 2004] or based on other identifying information within the packets (e.g., a Web cookie). The responder does not collaborate with other attackers but it tracks the identity of nodes with which it communicates and cannot be trusted. In each round, each node

Table I. The Probability of Success and the Number of Rounds for a Successful Predecessor Attack Against Various Defenses in the Static Membership Model

Defense Technique	Prob. of Success	If successful, rounds req., expectation	If successful, rounds req., Bounded with high probability
Static node membership model:			
Onion routing 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	$(\frac{c}{n})^2$	1	1

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 evaluate what I can do by only modifying its own probability of appearing in the path. Then, we evaluate *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. This attack could be based on packet counting [Serjantov and Sewell 2003] or on timing analysis [Danezis 2004; Levine et al. 2004]. 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 (otherwise, the system exposes new and returning users to immediate identification). 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.

A summary of our results is presented in Table I.

3.2 Initiators Kept Off Their Own Path

An initiator can attempt to reduce the number of times attackers observe her communication by ensuring that she does not appear in her own path of proxies, except as the first node, which is unavoidable. Unfortunately, unless the total number of nodes in the network is very small, the path lengths required will be very large. This is due to the initiator always being on the path; another node J would need to appear on the path nearly as often to fool the attacker.

Even if the path length is sufficiently long to delay the attack, the responder (or an eavesdropper) could monitor which nodes contact it in each round.

Eventually, every other node in the network will contact the responder. When this occurs, the initiator must be the only other node in the network that has not contacted the responder directly.

3.3 Fixed Nodes

We now turn to a defense based on the idea of keeping a single node at the front of the path, instead of selecting the node for that position randomly with each path change.

Freedman and Morris [2002] state that fixing nodes in this way provides greater deniability to any initiators linked to specific messages or responders by the predecessor attack. A user can claim that another user was selecting her first, and there would be no logs to disprove such a claim. Here, we extend this insight to a defense for both P2P anonymity systems and server-based systems like Tor.

The first node on the path must be stable. A node that is fixed in a position on the initiator's path and goes down regularly will cause periods of unavailability for the initiator or force the initiator to choose a new node to fix in the first position. We examine this assumption more closely in Section 4.

We now analyze this technique of selecting the same first node repeatedly. Let us assume that the nodes for all other positions are selected uniformly at random from all of N. Additionally, the first node is also selected uniformly from all of N at the creation of the first path. Let us call the node selected for this purpose the *helper node*, H. This defense protects the initiator from ever being tracked by the attacker, except if the node picked as H is an attacker node or is compromised by the 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. 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 $(c-1)/(n-1)$ chance in each round that the node selected is an attacker. Note that the initiator I should never select H for the last position, as it allows H, by itself, to immediately expose I as the initiator. Thus, in an expected $(n-1)/(c-1)$ round, I will be exposed as the initiator.

Since the probability of H being an attacker is $c/(n-1)$, 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 toward 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, toward one. We study the effect of node instability on this approach in Section 4.

Note that one could imagine similar approaches that include keeping the last node on the path the same. However, unlike the fixed first node defense, fixing the last position adds a considerable new danger. One of the goals of anonymity protocols is to protect against multiple responders that link user behavior. When fixing the last position, the last node on the path can be used as an identifier, just like the user's original IP address had been without an anonymous proxy. An additional proxy at the end of the path could be used to help hide this last identifier, but this merely pushes the problem back one step. In most cases this is an unacceptable risk, but it may be tolerable in a threat model that focuses on keeping the initiator's IP address hidden, rather than keeping the user activity unlinked.

4. ATTACKS AGAINST DYNAMIC SYSTEMS

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

This model attempts to capture the consequences of having users that stop making communications or stop the peer service on their local nodes. This occurs if users do not leave their systems turned on and connected to the network all the time. It is not clear that users of anonymous communications systems would leave their systems on and available for long, uninterrupted periods. To further examine this issue, we used traces of the uptimes of Tor network [Dingledine et al. 2004] proxies. In the last part of this section, we use the traces to compute the effectiveness of the intersection attack.

4.1 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. This technique is called the *intersection attack* [Raymond 2001]. The attack is well-known and there are few known means of defending against it. The designers of the Java Anon Proxy incorporated an "anonym-o-meter" to show users their current level of vulnerability to this attack [Köpsellbibt 2003; Berthold et al. 2000].[1]

Some peer-to-peer anonymous communications systems, including Tarzan and Crowds, are particularly vulnerable to this attack. In these two protocols, each participating peer knows the other available peers' IP addresses. This

[1]The anonym-o-meter does not correctly gauge this vulnerability: the implementation will sometimes report an increase in the anonymity level, but the nature of intersection makes it impossible for the anonymity to ever grow.

is critical for proper operation, because peers communicate directly in these protocols. In describing Tarzan, Freedman and Morris [2002] argue that having an incomplete list leaves the initiator vulnerable to other attacks in the peer selection process. 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 lists of peers in each round. For every round in which the initiator contacts the responder, the attacker can add that list of peers to her collection of sets of nodes. As nodes leave 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 few remaining nodes or use a denial of service attack to see if the traffic continues to the responder. The predecessor attack may be used simultaneously to identify likely initiators, further reducing the possibilities.

4.1.1 *Analysis.* This section shows that the intersection attack works efficiently in this dynamic model. Section 4.1.2 discusses other assumptions of this model in more detail.

We model the overall system with a discrete time model, with each time step being a round. We do not use a continuous-time model because we do not model the propogation of information about which nodes are in the system. We only know that nodes must have a consistent picture of which nodes are in the system when paths are constructed, which occurs before each round.

We model the uptime of users first with an exponential distribution and then a Pareto distribution. Pareto distributions have heavy-tailed behavior, and they can be parameterized to correspond roughly with the uptime of nodes observed in peer-to-peer file-sharing systems [Chu et al. 2002]. The Pareto distributions we use model some nodes remaining active for very long periods, while most nodes join the system for only a short time.

We do not model nodes that leave and join the system in the same round (i.e., if they become available in the next round) as having left at all. We construct our distributions such that such short-term failures are not considered events. Note that nodes newly joining the system do not affect the intersection attack, nor do nodes that intermittently rejoin. New nodes were not in the original anonymity set, and nodes that rejoin have already been eliminated from the anonymity set, so the attacker can ignore these nodes.

In the exponential model, we will say that the average uptime for a user is $1/\lambda$. 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 $F(T) = 1-e^{-\lambda T}$. We model the leaving and joining behavior of nodes as independent and iden-

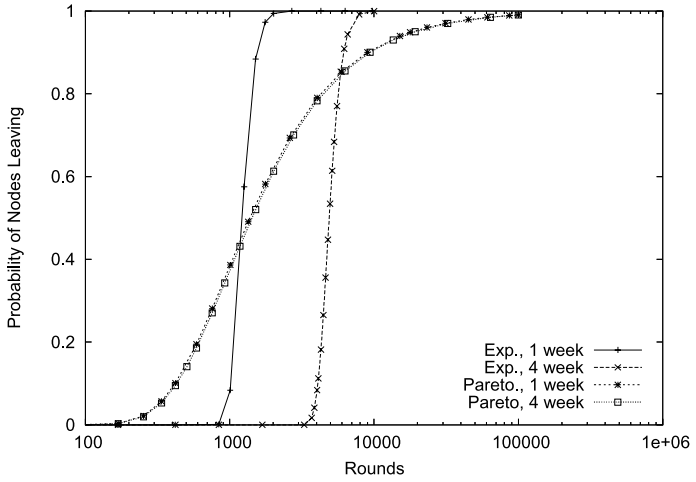

Fig. 1. Dynamic membership model: the intersection attack. The probability, for $n = 1,000$ nodes, of all nodes other than the initiator being intersected out of the set of possible initiators.

tically 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 $T = -1/\lambda \ln\left(1 - p^{1/(n-1)}\right)$. Note that T is linearly dependent on the average uptime, $1/\lambda$.

As we see in Figure 1, the probability of all nodes leaving rises quickly to one. The figure shows curves for the exponential distribution when the average uptime is one week and four weeks, i.e., 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. For this reason, the round length must be significantly less than the average uptime for the analysis to hold. The figure shows 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 probability 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 $E[T] = (ax_m)/(a-1)$, where x_m is the minimum uptime and where parameter a can be set given the desired average time. We chose to use $E[T_1] = 168$ hours and $E[T_4] = 672$ hours, i.e., one week and four weeks. The probability that a node leaves before time T is $F(T) = 1-(x_m/T)^a$. Again, we model nodes as i.i.d. processes, and the probability that all $n-1$ nodes leave by time T is given by $P(T) = \left(1 - (x_m/T)^a\right)^{n-1}$.

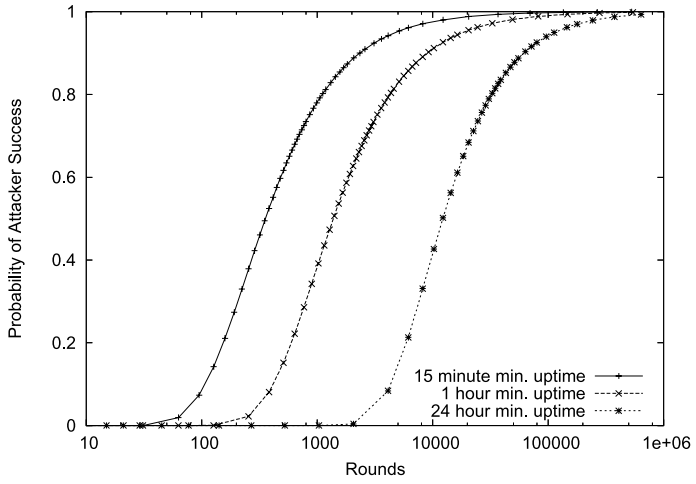

Fig. 2. Dynamic membership model: the probability that all nodes except the initiator are intersected out of the set of possible initiators based on a Pareto distribution of uptimes, starting with $n = 1000$ nodes, for different values of x_m, the minimum uptime.

To find the number of rounds the attacker requires to expose the initiator with probability $P(T) = p$, we can rearrange the formula:

$$T = \left(\frac{x_m}{1 - p^{1/(n-1)}} \right)^{1/a}$$

From Figure 1, when $x_m = 1$, 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. For example, for one-week average uptimes, the attacker requires 9,000 rounds before getting a 0.9 probability of identifying the initiator. This is because of 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, we can see that the average uptime makes little difference for the averages we used. This is reflected in the exponent in the formula for T, as the value of a does not change greatly between the one- and four-week average uptimes. Intuitively, both cases feature a few nodes that remain for long periods of time, far longer than the average case. Only with much shorter average uptimes do we see a significantly smaller time to attacker success. For example, with an average uptime of 16.8 hours, one tenth of a week, the attacker can identify the initiator with 0.9 probability in 5500 rounds, a 39% decrease over week-long average uptimes. This would require, however, very unreliable participants.

Since reasonable average uptimes do not impact the system, we also examine the effect of changing x_m, the minimum uptime in the system. As we see in Figure 2 for an average uptime of 168 hours, the minimum uptime makes

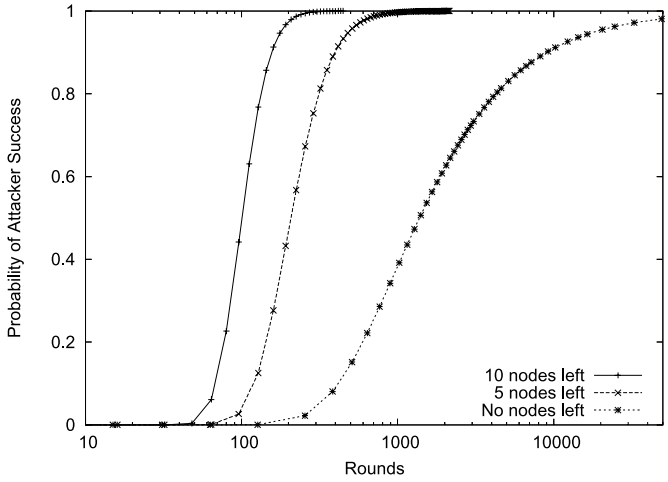

Fig. 3. Dynamic membership model: the probability of having less than ten and less than five nodes other than the initiator remaining in the system, and the probability of having no other nodes remaining, over time, where the system starts with $n = 1,000$ nodes.

a large difference in the time to attacker success. With a minimum uptime of 15 minutes, the number of rounds required to reach a 0.9 probability of attacker success is only 2,350, or about a quarter of that given by hour-long minimum uptimes. With a 24-hour minimum uptime, the attacker needs more than 60,000 rounds to attain the same 0.9 chance of success. Peer-to-peer environments would likely see very short minimum uptimes, but it is unclear if the model is realistic for all values x_m; as an example system, we examine the real uptimes of Tor nodes later in this section.

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 $\sigma = \left(1 - (x_m/T)^a\right)$.

In Figure 3, 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, using a minimum uptime of one hour. 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 and the minimum uptime is one hour. In the same two weeks, the attacker can get a list of five or fewer nodes with probability 0.828. We continue to see the effect of changing the minimum uptime in Figure 4.

4.1.2 *Caveats.* The intersection attack does not reveal the initiator in all situations, because the technique requires that the communication between

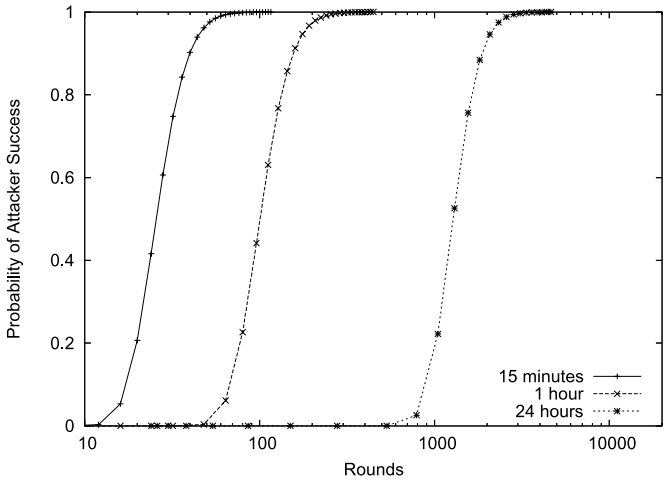

Fig. 4. Dynamic membership model: for different values of the minimal uptime, x_m, the probability of having less than ten and less than five nodes other than the initiator remaining in the system, and the probability of having no other nodes remaining, over time, where the system starts with $n = 1,000$ nodes.

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 placing more corrupt 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.

4.2 Fixed Node Defenses in the Dynamic Model

In Section 3, we described a method in which an initiator selects a node to permanently occupy the first position on its path. This allows the initiator to keep attackers from linking it to any responder, as long as at least the selected nodes were not an attacker. Here we give analysis for fixing the first node in the dynamic model.

This defense has different properties in the dynamic model as it depends on the stability of nodes in the system. Specifically, it requires the helper node to remain in the system as long as the initiator continues to contact a given

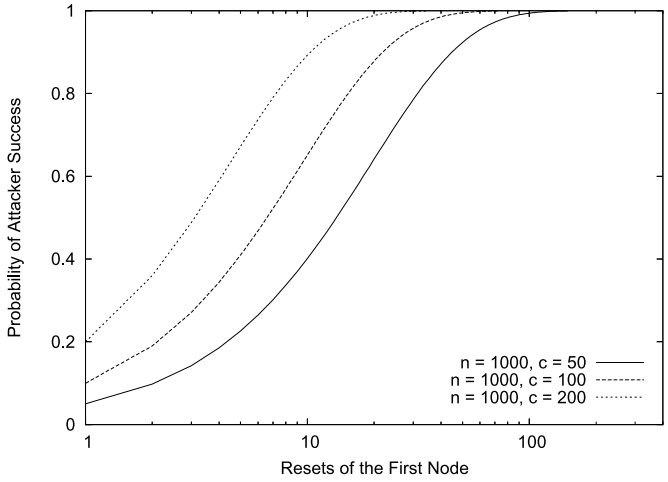

Fig. 5. Dynamic Model: probability of attackers' success when the first node is fixed. The time is given in the number of resets needed. We show results for $n = 1,000$ nodes, and $c = 50$, $c = 100$, and $c = 200$ attackers.

responder. If the nodes leave, the initiator must select new nodes to take their place. Every replacement of the nodes increases the chances that an attacker holds both positions simultaneously. The probability of exposure, then, grows closer to one over time.

Let us assume that nodes leave the network after a fixed, constant amount of time. In order to make comparisons with analytical results from prior work [Wright et al. 2002] and with simulation results from Section 5, we measure time in rounds. Let us say that a node remains in the system for ρ rounds, which we keep constant for simplicity. Also suppose that all nodes join the system at the same time. Then it is as if the system *resets* and forces a new path selection every ρ rounds.

For simplicity, we say that attacker nodes never leave the system. Thus, whenever an attacker is selected for the first node in the path, resets end and the attacker will eventually get enough data to determine the initiator.

When a new first node is selected, uniformly at random from N, there is a $\frac{c}{n}$ chance that the new node is an attacker. We want to know the number of resets, R, such that the probability of attacker success, P_R, reaches a given value p. Since $P_R = 1 - (1 - P_1)^R$, substituting values of P_1 gives $R = \frac{\ln(1-P_R)}{\ln(\frac{n-c}{n})}$.

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 II).

In Figure 5, we see that the probability of the attackers succeeding grows toward one as the number of resets increases. The similarity to simulation results for the predecessor attack in Section 5, when paths are selected uniformly at random, suggests that fixing the first and last nodes may not provide significantly stronger security over time when nodes leave the system frequently.

Table II. Number of Rounds for a Successful Predecessor Attack for Various Defenses

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

This is in contrast with the analysis for the static model given in Section 3.3, in which fixing the first node appears to be an effective 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.

4.2.1 *Discussion.* There exists an alternative when nodes leave but later return to the system. The initiator could use one node as its only first node. If the node was unavailable, it would not communicate with the responder.

This approach is vulnerable, however, to denial-of-service attacks. To continue communicating with the responder even when the selected node is unavailable, a backup node could be kept. In general, the initiator could keep a small pool of nodes from which it selects the first node on its path. As long as at least one node in the pool is available, the initiator can communicate with the responder.

In fact, the current Tor specification uses a fixed first node, which is called a *guard node*. Each Tor node maintains an ordered list of such nodes, adding replacements for unavailable guard nodes at the end of the list. The specification states that each time a path is created, nodes chose a guard node randomly from among the first three usable guards.

There is a tradeoff in this scheme between security and usability. To prevent a lack of service, the initiator must keep a sufficiently large pool to ensure that a node is available at all times. But the larger the pool, the greater the chance that a attacker node is in the pool. When there are attacker nodes in the pool, it is very likely that such a node will be chosen as the first node many times over the user's network lifetime. This is especially true if attackers, seeking to increase their chances of being selected, remain online a larger percentage of the time than other nodes.

In particular, for Tor's method of choosing among three guard nodes for each path, the chance that attackers are are the beginning and end of the path is $\frac{c}{n}(1-\frac{n-c}{n}\cdot\frac{n-c-1}{n-1}\cdot\frac{n-c-2}{n-2})$ as nodes for the pool are chosen without replacement. This is somewhat less secure than a probability of $(\frac{c}{n})^2$ when the same guard is always used. For example, when there are $n = 1{,}000$ nodes in the system, and the fraction of attacker nodes is 10%, the probability of attacker success is 2.7%, rather than 1.0%; when a third of the network are attackers, the probability is 25% rather than 11%. However, Tor users receive better performance since a given guard node may provide poor service. Moreover, no single guard node sees all traffic that originates from the user sent to the Tor network.

4.3 Trace-Driven Evaluation

To further investigate the efficacy of the intersection attack, we acquired traces of the Tor anonymous communications system. Although this is a server-based system, rather than a peer-to-peer system, the server operators are volunteers and service is not guaranteed in any way. Thus, we can expect some down time for maintainance and other reasons, although perhaps at a lower rate than in a peer-to-peer system. These traces should therefore be conservative with respect to the efficacy of intersection attacks.

The traces record the uptimes of proxies during a 446-day period, from November 10, 2004, to January 29, 2006. Below, we characterize what the traces tell us about the Tor network, and then we show our results of simulating an intersection attack.

4.3.1 *Characteristics of the Tor Network.* The logs are the result of the efforts of the Tor network creators Roger Dingledine, Nick Mathewson, and Paul Syverson, as well as the early server operators. The network is quite an achievement. The logs are archived at http://www.noreply.org by Peter Palfrader, who generously allowed us access to the raw files.

The 3,911 log files are each four hours apart. Among other information, each contains the uptime of each proxy as reported by the proxy itself to the logging server. We had to infer the downtimes of proxies by parsing the logs. If a proxy reported an uptime less than their last log entry, we assume the proxy was down from the last reported entry until the next entry (minus the new uptime). In the worst case, the proxy could have rebooted just after reporting an uptime and then four hours later, rebooted again just before reporting a new uptime. We have no method of determining how often this occurred.

A small number of proxies erroneously reported the time since the UNIX epoch as their uptime and we discounted all logs from such proxies. Additionally, we ignore partially observed data, instead focusing on the distribution created from complete duty cycles.

Figure 6 plots the number of active proxies according to the logs for the 446-day period. Figures 7a and 7b show the observed PMFs of the uptime and downtime of peers, respectively. There are a few characteristics to note about Tor proxies. First, the popularity of the Tor network is steadily growing. Second, the uptimes of proxies are very similar to measurements of other peer-to-peer networks. Proxy uptimes have a median length of 6.3 hours, 80% last no more than 22 hours, and 99% are under 13 days. Studies of peer-to-peer file-sharing systems show that most users of these systems also do not stay connected for long periods of time. Our previous work [Bellissimo et al. 2004] measured the session lengths of BitTorrent peers. We found that median session length was 7.9 hours, 80% lasted no more than 24 hours, and 99% of all sessions were shorter than five days. A study by Sariou et al. [2002] found a median session time of only 60 minutes.

It is difficult to say that Figure 7a clearly matches a known distribution, such as the exponential or Pareto distributions we use earlier in this section. Log normal, Weibull, and a mixture model of two log normals were not high-quality fits either (results are not shown). This is likely because there is not

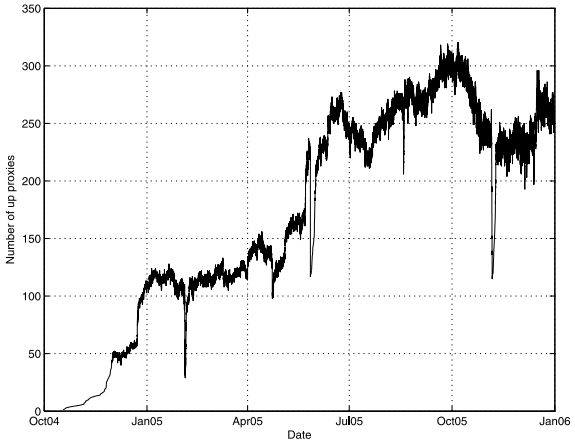

Fig. 6. Tor Network: the number of active proxies in the Tor network from November 2004 to January 2006.

one simple factor that determines the uptimes of most nodes. For example, we believe dips in Figure 6 are due to time spent reconfiguring new versions of the Tor software.

4.3.2 *Evaluation.* We used the traces to perform an intersection attack on the Tor network, modeled as a peer-to-peer system. Our evaluations represent a worst-case scenario for the initiator. We assume the initiator contacts the responder every round, and we assume the attacker knows this. With such assumptions, attackers need only keep track of Tor membership, which is public knowledge so that paths can be formed.

In our simulation, we ran 446 trials, one for each day of the log period. For each trial, the attacker noted the initial group of proxies and then observed when a proxy was absent for an entire round.

Figure 8a shows the success of the attack for different round lengths, with the number of days since the start of the session on the x-axis. Longer rounds make the attack more difficult since nodes must appear in only part of the round to thwart the attacker. Figure 8b shows the same data in the format of Figures 1 and 2, for comparison. The plots represent the mean values of all trials contributing to the round or day on the x-axis. The standard deviations of these means were very small and are not shown for clarity.

We can see that most nodes lose about half their protection well before the first ten rounds, which is within the first several days. Unfortunately, the remaining anonymity degrades quickly as well. This experiment suggests that Tor is quite fragile. In practice, the attack may not succeed as quickly for two reasons.

The first reason is that the attacker may not know in exactly which rounds the responder is contacted. For Crowd-based sessions, the attacker must only appear as one of the proxies. For a probability of forwarding, $0.5 < p < 1$, this will occur with probability $(1 - \frac{c}{N})^{1/p}$, for a proxy set of size N containing c attackers. For other protocols, appearing as the tail node occurs with the much lower probability of $\frac{c}{N}$.

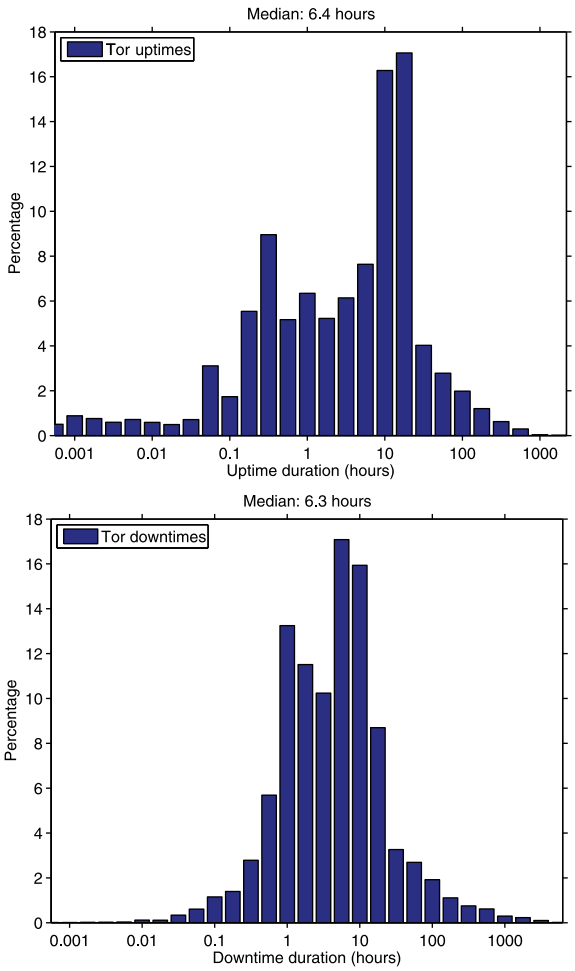

Fig. 7. Tor uptime and downtime: note that the graphs are on a log scale with exponentially sized bins.

However, the attacker may be the responder. Alternatively, the attacker may collude with the responder, eavesdrop on the responder's connection to the Internet, or compromise the responder's computer to learn of incoming connections.

The second reason is that the initiator may not contact the responder every round. This is the more likely mitigating factor for the duration of the attack. Section 6 discusses this possibility further in the context of the predecessor attack; those results generally apply to the intersection attack as well.

5. SIMULATING THE PREDECESSOR ATTACK

In our prior work, we provided upper bounds on the number of rounds required for attackers to perform the predecessor attack against nodes running Crowds,

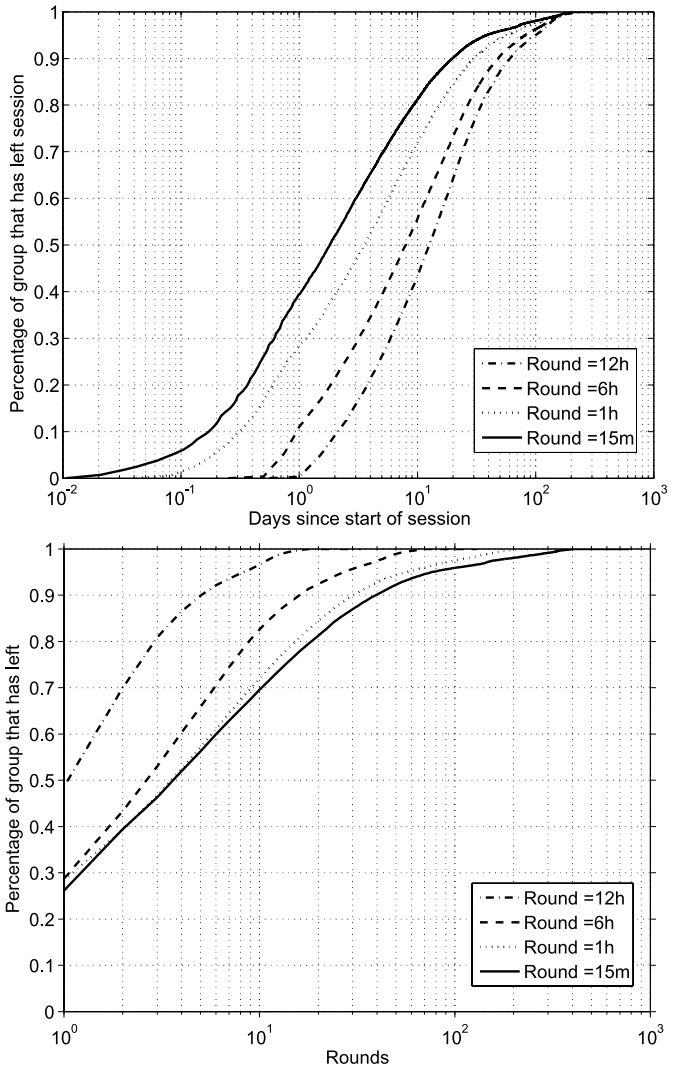

Fig. 8. The intersection attack: the probability, for the Tor network, of all nodes other than the initiator being intersected out of the set of possible initiators. The lower graph is for comparison against Figs. 1 and 2.

Onion Routing, and Mix-Nets. While these bounds helped to indicate the relative performance of each protocol against the attacks, they are asymptotic and do not provide a tight window on the rounds the attacker would actually need.[2]

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 the methods for selecting fixed first nodes in

[2]Bounds tighter than our results from prior work were given by Figueiredo et al. [2004].

Onion Routing paths described in Section 3 and compare those results against uniform path selection.

5.1 Simulation Methodology

We simulated long-running Crowds-based and Onion Routing-based sessions. We chose these protocols as models of open systems with relatively low overhead. With paths of length five and greater, the expected time to attack Mix-Nets also falls far outside the times we see in these simulations. Furthermore, the idealized Mix-Nets that we studied in our prior work [Wright et al. 2004] do not appear to be reasonable for low-latency systems due to timing analysis attacks [Danezis 2004; Levine et al. 2004]. High-latency message systems like Mixminion [Danezis et al. 2003], which might be modeled as ideal mixes, tend to be more difficult to attack with the predecessor attack, as it takes more effort to profile the user. Thus, we restrict our simulation study to Crowds and Onion Routing systems, both of which are clearly vulnerable to the predecessor attack with reasonable attacker resources.

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). Below, we provide the attacker algorithm. This algorithm differs from the more easily analyzed attacker algorithm used in our prior work, which may explain some of the disparity between our results here and the bounds.

Let us define N as the set of n total nodes, C as the set of c attackers, an attack server S, and a number of rounds T that the attack runs. We define an attack more precisely as follows. Let the times at which packets are sent by node X on J's path P_J, be $Sent(X, P_J)$. Let the times at which packets are received by node Y on node J's path P_J be $Recv(Y, P_J)$. Let $Timing()$ be a function of a set of sent times and a set of received times. We define the timing analysis function as: $Timing(Sent(M_1, P_J), Recv(M_2, P_K)) = True$ iff $P_J = P_K$ and M_1 appears on P_J before M_2.

ATTACKER ALGORITHM 5.1

(1) Add $c \geq 2$ attackers (the set of attackers C) into an Onion Routing system with $n - c$ nodes. The set N now consists of n nodes. Attackers operate according to the system protocol, i.e., either Onion Routing or Crowds as appropriate.
(2) S initializes a counter to zero for each non-attacker node J in the system; i.e., $\forall J \in (N - C)$, $Counter(J) = 0$.
(3) In each round T_i, attackers and S follow either Algorithm 5.1 for Crowds or Algorithm 5.1 for Onion Routing, in which S updates the counters for each node.
(4) Repeat Steps 2–3 for T rounds. Output the node K such that $\forall J \in (N - C)$, $Counter(K) \geq Counter(J)$. Break ties by random selection.

ATTACKER ALGORITHM 5.2

(1) For each attacker $X \in C$, for each path P on which X appears, let $J \in (N - C)$ be the node preceeding X on P. If X observes a connection to R using P, X sends a message to S with the IP address of J_P.
(2) S increments $Counter(J)$

ATTACKER ALGORITHM 5.3

(1) For each attacker $X \in C$, for each path P_J belonging to J on which X appears before the third to last node, log the times $Sent(X, P_J)$. At the end of the round, send $Sent(X, P_J)$ and the IP address of J to S.
(2) For each attacker $Y \in C$, for each path P_K belonging to node K and on which Y appears as the last node, log the times $Recv(Y, P_K)$. If the path was used to communicate with R, Y sends $Recv(Y, P_K)$ to S at the end of the round.
(3) If S does not receive a set of received times, $Recv(Y, P_K)$, stop.
(4) For each set of times sent, $Sent(X, P_J)$, if $Timing(Sent(X, P_J), Recv(Y, P_K)) = True$, S increments $Counter(J)$.

Algorithm 5.1 differs from those used in our prior work in that it outputs the most frequently counted node and does not require that the count exceed a threshold value [Wright et al. 2002]. Additionally, Algorithm 5.1 accounts for the fact that an attacker node near the end of the path can determine that it is not the first node on the path. This is because it is either communicating directly with the last node, which must be an attacker if it generates $Recv(Y, P_K)$, or share the same node between them if it is third to last. In the latter case, it is much more likely that the nodes are one hop apart than that the shared node is repeated on the path. Knowing this, the attackers will most likely risk a false negative due to a large chance of a false positive; this is the choice made in our attacker model.

Each data point in the graphs of our results in this section is based on the average of ten *runs*. Each run consists of 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. For each of the 10,000 simulations in a run, the attackers waited a given number of rounds and then made their guess of the initiator's identity based on their current logging information. We then determined the percentage of the 10,000 simulations in which the attackers guessed correctly. This approach provides the attackers with an average success rate after a given number of rounds.

Performing this ten times enabled us to give an average and determine variation in the results. In all, we ran more than 66.8 million simulations. For all graphs, we computed standard deviations, but they are not shown in most figures as they are too small to be visually significant (less than one percent for all data and less than one tenth of one percent for high probabilities of attacker success).

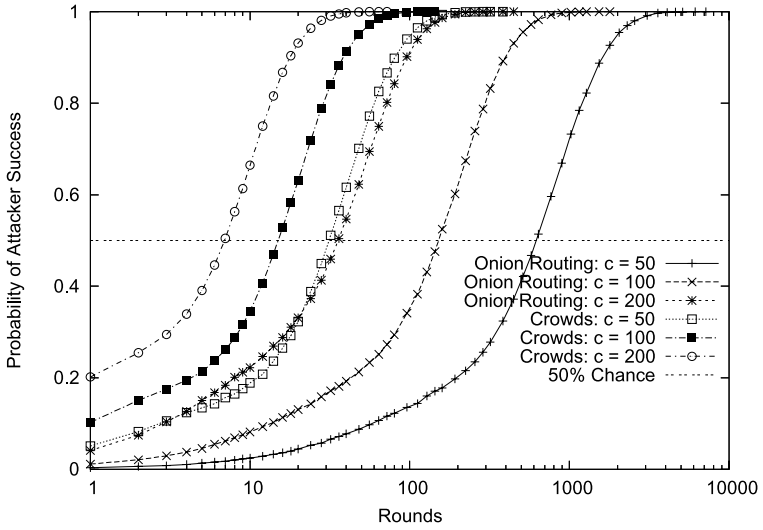

Fig. 9. Static membership model: simulation of the predecessor attack against Crowds and Onion Routing, for varying numbers of attackers.

5.2 Results

Figure 9 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, out of 10,000, in which the attacker correctly determined the initiator for a given number of rounds. In all simulations, there are $n = 1,000$ total nodes and the average path length is 10. We show a dotted-line at $y = 0.50$, as that is the dividing line between *Probable Innocence*, where the attackers' guesses are more than 50% likely to be wrong, and *Possible Innocence*, where the attackers guesses are more than 50% likely to be correct [Reiter and Rubin 1998]. Although this line is arbitrary, it represents a point at which the initiator's anonymity is compromised beyond an acceptable level, although the attackers cannot yet be highly confident in their guess.

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 [Reiter and Rubin 1998], that gave the same average path length.

From Figure 9, 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 $n/c = 10$, the attackers require only about 15 rounds against Crowds to get a 50% chance of exposing the

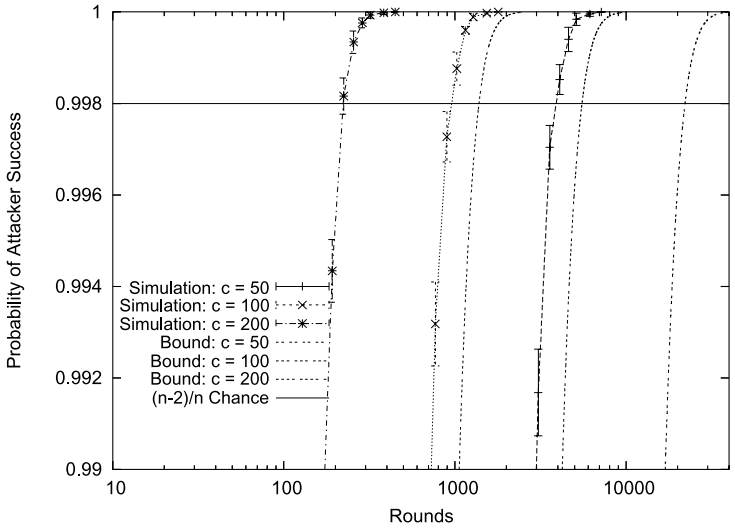

Fig. 10. Static membership model: the predecessor attack against Onion Routing, top 1%.

initiator. For the same value of $c = 100$ in Onion Routing, the attackers require approximately 150 rounds. This ten-fold increase in the number of rounds appears to match the additional factor of n/c found in the analytical results from our prior work [Wright et al. 2002]. This trend can be seen throughout the graph.

We also see from Figure 9 that the larger the value of n/c, the longer the predecessor attack takes. For example, with Onion Routing, $c = 50$ attackers require over 600 rounds to find the initiator with 50% or better probability. This is about four times longer than for $c = 100$, for which 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 n/c. This dependence holds for most values, but the differences are greater for low numbers of rounds. The linear dependence on c/n for Crowds also holds for most values, again excepting the lowest counts.

Figure 10 shows the predecessor attack against Onion Routing as in Figure 9, 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 $(n - 2)/n$, which is the standard of high probability we set in the analysis from our prior work [Wright et al. 2002]. To obtain a line, rather than a high-probability bound, we used the same Chernoff bound equations, where we fix the number of rounds as the x-axis value and obtain the probability value for the y-axis. Thus the entire line serves as an upper bound on the number of rounds required for the attacker to reach the given rate of success.

From the figure, we see that the number of rounds guaranteed by the analytic bounds are much higher than the simulation results. The number of rounds required for the attackers to be logged with high probability $(n - 2)/n$

is between 5.7 and 6.2 times less for the simulation than guaranteed by the bounds from our prior work [Wright et al. 2002].

We also see from the figure that the relationships between the numbers of rounds required for different values of 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 $(n-2)/n$ 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 11 with a fixed ratio of $n/c = 10$ 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 $n/c = 10$, as calculated from the formula below, where Event 5.1 is the event that the initiator is logged as the predecessor in a given round:

Event 5.1. In a given round, S increments $Counter(I)$.

The formula is derived in our prior work using a Chernoff bound [Wright et al. 2002].

$$\begin{aligned} Pr\{B(T, Pr\{Event\ 5.1\}) \leq (1-\delta)TPr\{Event\ 5.1\}\} &< e^{-1/2\delta^2 TPr\{Event\ 5.1\}} \\ &< e^{-T(1/8)(c/n)^2} \end{aligned} \quad (1)$$

As in Figure 10, we see the significant difference between the predicted values of the bounds and the simulation results. We can also see the small, 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 bound and is not reflected in the insignificant difference in the results between $n = 1,000$ and $n = 10,000$.

The explanation for this effect is that, when there are fewer nodes to select from, a node may be chosen more frequently to be on the initiator's path and may then be logged more often by the attackers. With more nodes, each selected uniformly at random, the chances of this are lower, so the attack largely depends on logging the initiator multiple times. The bound does not show this effect; there is no dependence on n if the ratio n/c is fixed.

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 12.

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 12, we see that average path length is more significant before the probability of attacker success has reached 50% than afterwards. Also, the

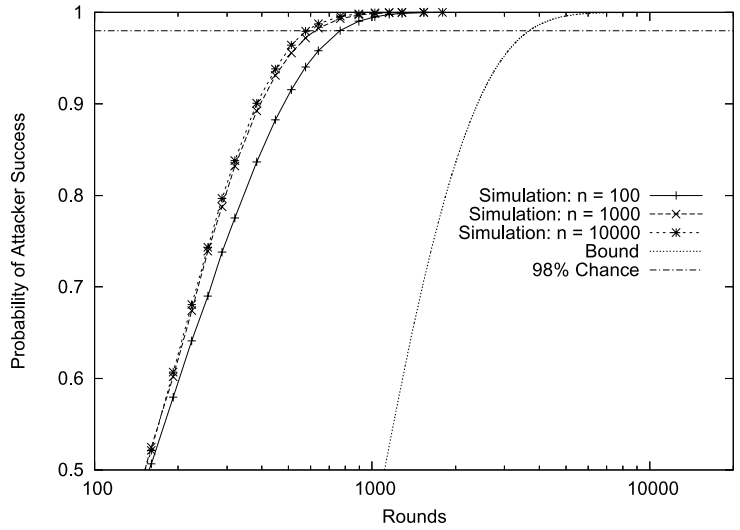

Fig. 11. Static Model: the predecessor attack against Onion Routing, for varying n, top 50%.

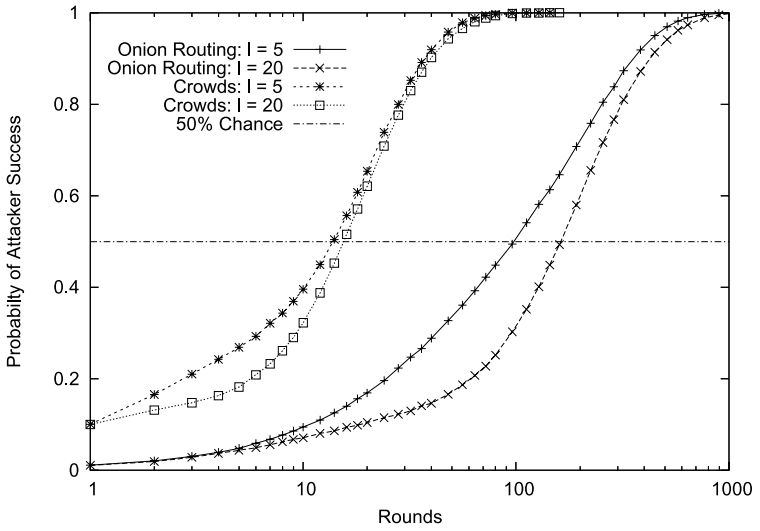

Fig. 12. Static Model: the predecessor attack against Onion Routing and Crowds, for varying average path lengths.

difference between different path lengths is larger for Onion Routing, while largely insignificant for Crowds.

6. 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

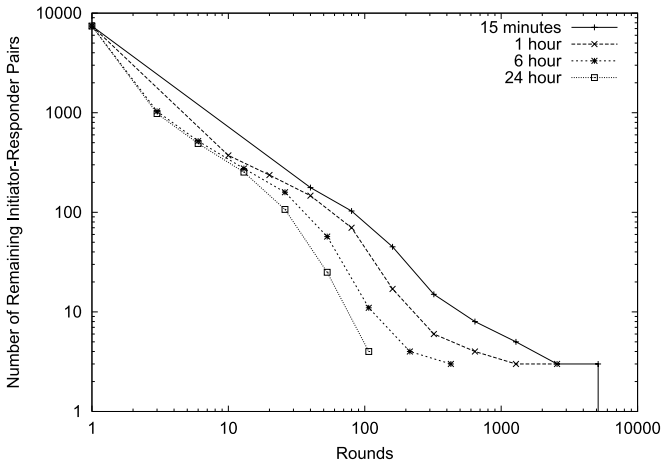

Fig. 13. Survivor plot of users contacting the same Web site over increasing numbers of rounds, from data collected at the University of Massachusetts.

real users exhibit such behavior, we obtained traces of communications from a Web proxy in the Department of Computer Science at the University of Massachusetts Amherst. The data was collected by Diaz and Allan [2003] from June 3 to October 28, 2002, from 17 University of Massachusetts volunteers (see Figure 16). We also analyzed data collected previously at the University of California at Berkeley [Gribble 1997].

We sought two important results from the data. First, we wanted to know the frequency that users contact the same Web site (i.e., the responder) and the extent to which that contact continues to occur. It is critical to note that the predecessor attack works in rounds, regardless of their 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.

One could argue that users of anonymous communications systems would be more careful to not behave in trackable ways than the volunteers of our study. We believe that this is unlikely. The users of our study knew that their Web usage was being tracked in a linkable way. Users of anonymous communications systems might be inclined to behave cautiously, but would have reason to believe that their patterns of communication are mostly hidden. Some of these users might even seek systems for anonymity with the intention of using them for repeated communications to a specific responder or set of responders, rather than for general privacy.

In any case, users should know what patterns of communications can be tracked in order to avoid them when communicating anonymously.

Figure 13 tracks whether users in the study revisited the same responder over time, using the data collected at the University of Massachusetts. If we

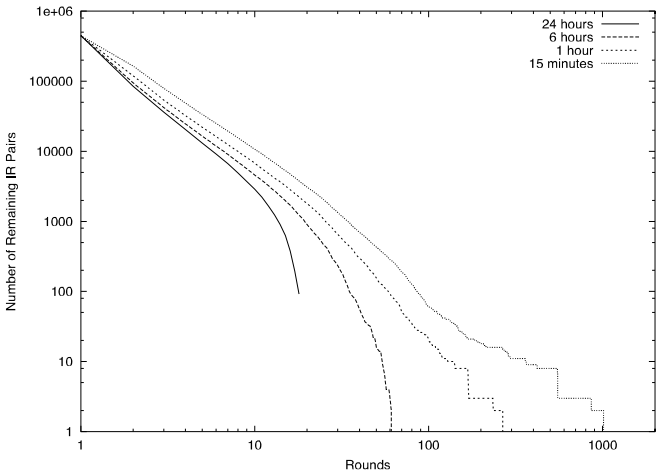

Fig. 14. Survivor plot of users contacting the same Web site over increasing numbers of rounds, from data collected at the University of California at Berkeley.

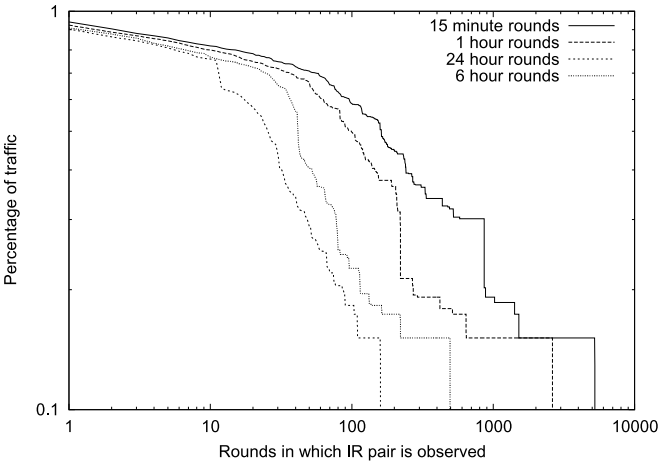

Fig. 15. Percentage of all traffic represented by initiator-responder pairs observed in varying numbers of rounds, from data collected at the University of Massachusetts.

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 13. We obtained similar results from analyzing 18 days' worth of data collected from the University of California at Berkeley's HomeIP service, as shown in Figure 14.

Only a small percentage of connections are long lasting, but these are a signficant part of the users' behavior. Figure 15 shows what percentage of all traffic is represented by long-lasting sessions. The longest lasting sessions

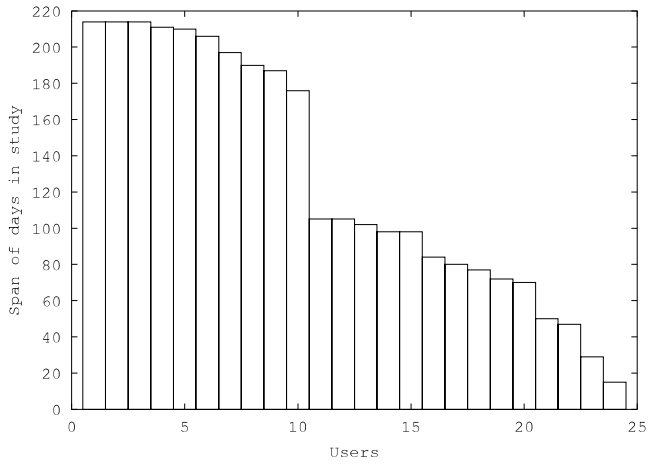

Fig. 16. Length of time each user employed the Web proxy.

make up approximately 10% of all traffic, representing a significant fraction of users' behavior.

In our simulation results from Section 5, we found that in an Onion Routing system with 1,000 nodes 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. There were five connections seen in more than 1,284 15-minute rounds and would therefore be trackable with high probability during this period. 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 period. These connections make up 36% of all the users' traffic. Thus, over one-third of the traffic that went through the Web proxy faced an 80% chance of linkage to the initiator.

7. CONCLUSION

In this article, we explored the consequences of modifying and validating the assumptions of our model from prior work. We first modified our assumptions by allowing users to select nodes in a non-uniform manner. This gave rise to new defenses that can reduce the possibility of performing the predecessor attack.

The next modification was to remove the assumption of a static membership set. By studying dynamic membership, we learned that the attacker gains additional leverage to degrade the anonymity of initiators who continue to use the system. Specifically, fixed-node defenses can be neutralized and peer-to-peer protocols, including Tarzan and Crowds, are particularly vulnerable to the intersection attack. Note that other peer-to-peer protocols, such as MorphMix, are not as vulnerable to this attack, since awareness of all other peers is not needed for operation of the protocol.

Next, we tested the analytic results in simulation. In doing this we found that the attacker could succeed in much less time than given by the analytic upper bounds. At the same time, the simulation results validated the relationships between different scenarios that are predicted by the bounds.

Another test was to validate the assumption that users have trackable patterns of use. We studied real network traffic and found that a substantial fraction of users follow the necessary communication patterns for the attacks 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; users of current protocols need to be cautious in how often and how long they attempt to communicate anonymously. When users and servers often leave and join the system, the system is particularly vulnerable to attack. However, defenses such as the fixed first node defense remain useful in relatively static environments.

ACKNOWLEDGMENTS

We are grateful to Peter Palfrader for providing a copy of Tor log files, to Roger Dingledine and Paul Syverson for clarifying details of the Tor operation and design decisions, and to Fernando Diaz and James Allan for providing access to their traces of Web access.

REFERENCES

BÄCHER, P., HOLZ, T., KÖTTER, M., AND WICHERSKI, G. 2005. Know your enemy: Tracking botnets. http://www.honeynet.org/papers/bots. The Honeynet Project and Research Alliance.

BACK, A., GOLDBERG, I., AND SHOSTACK, A. 2000. Freedom 2.0 security issues and analysis. White paper. Zero-Knowledge Systems, Inc.

BARYSHNIKOV, Y., COFFMAN, E., PIERRE, G., RUBENSTEIN, D., SQUILLANTE, M., AND YIMWADSANA, T. 2005. Predictability of Web-Server traffic congestion. In *Proceedings of the International Workshop on Web Content Caching and Distribution (WCW'05)*. 97–103.

BAUER, K., MCCOY, D., GRUNWALD, D., KOHNO, T., AND SICKER, D. 2007. Low-resource routing attacks against anonymous systems. Tech. rep. CU-CS-1025-07, University of Colorado at Boulder.

BELLISSIMO, A., SHENOY, P., AND LEVINE, B. N. 2004. Exploring the use of BitTorrent as the basis for a large trace repository. Tech. rep. 04-41, Department of Computer Science, University of Massachusetts at Amherst.

BERTHOLD, O., FEDERRATH, H., AND KÖHNTOPP, M. 2000. Project Anonymity and unobservability in the Internet. In *Proceedings of Computers Freedom and Privacy Conference (CFP'00)*. 57–65.

BERTHOLD, O. AND LANGOS, H. 2002. Dummy traffic against long term intersection attacks. In *Proceedings of Workshop on Privacy Enhancing Technologies (PET'02)*. 110–128.

BISSIAS, G. D., LIBERATORE, M., AND LEVINE, B. N. 2005. Privacy vulnerabilities in encrypted HTTP streams. In *Proceedings of Workshop on Privacy Enhancing Technologies (PET'05)*. 1–11.

CHAUM, D. 1988. The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Crypto. 1,* 1, 65–75.

CHU, J., LABONTE, K., AND LEVINE, B. N. 2002. Availability and locality measurements of peer-to-peer file systems. In *Proceedings ITCom: Scalability and Traffic Control in IP Networks II Conference*. Vol. SPIE 4868. 310–321.

DANEZIS, G. 2003. Statistical disclosure attacks: Traffic confirmation in open environments. In *Proceedings of Security and Privacy in the Age of Uncertainty (SEC'03)*. 421–426.

DANEZIS, G. 2004. The Traffic analysis of continuous-time mixes. In *Proceedings Workshop on Privacy Enhancing Technologies (PET'04)*. 35–50.

DANEZIS, G., DINGLEDINE, R., AND MATHEWSON, N. 2003. Mixminion: design of a type III anonymous remailer protocol. In *Proceedings of the IEEE Symposium on Security and Privacy*. 2–15.

DAVISON, B. 2002. Predicting Web actions from HTML content. In *Proceedings of the ACM Conference on Hypertext and Hypermedia (HT'02)*. 159–168.

DÍAZ, C., SEYS, S., CLAESSENS, J., AND PRENEEL, B. 2002. Towards measuring anonymity. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET'02)*. 184–188.

DIAZ, F. AND ALLAN, J. 2003. Browsing-based user language models for information retrieval. Tech. rep. CIIR IR-279, University of Massachusetts at Amherst.

DINGLEDINE, R. AND MATHEWSON, N. 2007. Tor path specification. http://tor.eff.org/svn/trunk/doc/spec/path-spec.txt.

DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. 2004. TOR: The next-generation onion router. In *Proceedings of USENIX Security Symposium*. 303—320.

DOUCEUR, J. R. 2002. The Sybil attack. In *Revised Papers from the 1st International Workshop on Peer-to-Peer Systems*. Springer-Verlag, 251–260.

DUCHAMP, D. 1999. Prefetching hyperlinks. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*. 127–138.

FIGUEIREDO, D. R., NAIN, P., AND TOWSLEY, D. 2004. On the analysis of the predecessor attack on anonymous protocols. Tech. rep. 04-65, Department of Computer Science. University of Massachusetts.

FREEDMAN, M. AND MORRIS, R. 2002. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'02)*. 193–206.

GOLDBERG, I. AND WAGNER, D. 1998. TAZ servers and the rewebber network: Enabling anonymous publishing on the World Wide Web. *First Monday*.

GOLDSCHLAG, D., REED, M., AND SYVERSON, P. 1996. Hiding routing information. In *Proceedings of Information Hiding Workshop (IH'96)*. 137–150.

GRIBBLE, S. 1997. UC Berkeley home IP HTTP traces. http://www.acm.org/sigcomm/ITA/.

HINTZ, A. 2002. Fingerprinting websites using traffic analysis. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET'02)*. Springer-Verlag, Lecture Notes in Computer Science, vol. 2482, 229–233.

Honeynet Project 2005. Know your enemy: Tracking botnets – spreading. http://www.honeynet.org/papers/bots/botnet-spreading.html. The Honeynet Project and Research Alliance.

KESDOGAN, D., AGARWAL, D., AND PENZ, S. 2002. Limits of anonymity in open environments. In *Proceedings of Information Hiding, 5th International Wkshp (IH'02)*. 53—69.

KESDOGAN, D., EGNER, J., AND BÜSCHKES, R. 1998. Stop-and-go-MIXes providing probabilistic anonymity in an open system. In *Information Hiding*. Lecture Notes in Computer Science, vol. 1525. Springer, 83–98.

KÖPSELLBIBT, S. 2003. JAP — Web mixes. http://www.petworkshop.org/2003/slides/panels/stefan-PET2003-panel.pdf.

LEVINE, B., REITER, M., WANG, C., AND WRIGHT, M. 2004. Timing attacks in low-latency mix systems. In *Proceedings of Financial Cryptography (FC'04)*. (Lecture Notes in Computer Science, vol. 3110). 251—265.

LEVINE, B. AND SHIELDS, C. 2002. Hordes: A protocol for anonymous communication over the Internet. *ACM J. Comput. Secur. 10,* 3, 213–240.

LIBERATORE, M. AND LEVINE, B. N. 2006. Inferring the source of encrypted HTTP connections. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS'06)*. 255–263.

MARGOLIN, N. B. AND LEVINE, B. N. 2007. Informant: Detecting sybils using incentives. In *Proceedings of Financial Cryptography (FC'07)*.

MATHEWSON, N. AND DINGLEDINE, R. 2004. Practical traffic analysis: Extending and resisting statistical disclosure. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET'04)*. Lecture Notes in Computer Science, vol. 3424. 17–34.

MURDOCH, S. J. 2006. Hot or not: Revealing hidden services by their clock skew. In *Proceedings of the ACM Conference on Computer and Communications Security*. 27–36.

MURDOCH, S. J. AND DANEZIS, G. 2005. Low-cost traffic analysis of Tor. In *Proceedings of the IEEE Symposium on Security and Privacy*. 183–195.

ØVERLIER, L. AND SYVERSON, P. 2006. Locating hidden servers. In *Proceedings of the IEEE Symposium on Security and Privacy*. 100–114.

RAYMOND, J. F. 2001. Traffic analysis: Protocols, attacks, design issues and open problems. In *International Workshop on Design Issues in Anonymity and Unobservability*. Lecture Notes in Computer Science, vol. 2009. Springer, 10–29.

REITER, M. K. AND RUBIN, A. D. 1998. Crowds: Anonymity for Web transactions. *ACM Trans. Inform. Syst. Secur. 1,* 1, 66–92.

RENNHARD, M. AND PLATTNER, B. 2004. Practical anonymity for the masses with MorphMix. In *Proceedings of Financial Cryptography (FC'04)*. 233–250.

SAROIU, S., GUMMADI, P. K., AND GRIBBLE, S. 2002. A measurement study of peer-to-peer file sharing systems. In *Proceedings of the Multimedia Computing and Networking Conference (MMCN'02)*. 314–329.

SCARLATTA, V., LEVINE, B., AND SHIELDS, C. 2001. Responder anonymity and anonymous peer-to-peer file sharing. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP'01)*.

SERJANTOV, A. AND DANEZIS, G. 2002. Towards an information theoretic metric for anonymity. In *Proceedings of the Workshop on Privacy Enhancing Technologies (PET'02)*. Lecture Notes in Computer Science, vol. 2482. 259–263.

SERJANTOV, A. AND SEWELL, P. 2003. Passive attack analysis for connection-based anonymity systems. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS'03)*. 116–131.

SHERWOOD, R., BHATTACHARJEE, B., AND SRINIVASAN, A. 2005. P5: a protocol for scalable anonymous communication. *J. Comput. Secur. 13,* 6, 839–876.

SHMATIKOV, V. 2002. Probabilistic analysis of anonymity. In *IEEE Computer Security Foundations Workshop*. 119–128.

SUN, Q., SIMON, D. R., WANG, Y.-M., RUSSELL, W., PADMANABHAN, V. N., AND QIU, L. 2002. Statistical identification of encrypted Web browsing traffic. In *Proceedings of the IEEE Symposium on Security and Privacy*. 19–30.

SYVERSON, P., TSUDIK, G., REED, M., AND LANDWEHR, C. 2000. Towards an analysis of onion routing security. In *Proceedings Workshop on Design Issues in Anonymity and Unobservability*. Lecture Notes in Computer Science, vol. 2009. 96–114.

THE GRAPHIC, VISUALIZATION, AND USABILITY CENTER. 1998. GVU's 10th WWW User Survey. http://www-static.cc.gatech.edu/user_surveys/survey-1998-10.

WRIGHT, M., ADLER, M., LEVINE, B. N., AND SHIELDS, C. 2002. An analysis of the degradation of anonymous protocols. In *Proceedings of ISOC Network and Distributed System Security Symposium (NDSS'02)*. 38–50.

WRIGHT, M., ADLER, M., LEVINE, B. N., AND SHIELDS, C. 2003. Defending anonymous communication against passive logging attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*. 28–41.

WRIGHT, M., ADLER, M., LEVINE, B. N., AND SHIELDS, C. 2004. The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Trans. Inform. Syst. Secur. 4,* 7, 489–522.

Received April 2006; revised June 2007; accepted July 2007