Trust-based Anonymous Communication: Adversary Models and Routing Algorithms

Aaron Johnson* Paul Syverson
U.S. Naval Research Laboratory
{aaron.m.johnson,paul.syverson}@nrl.navy.mil

Roger Dingledine Nick Mathewson
The Tor Project
{arma, nickm}@torproject.org

ABSTRACT

We introduce a novel model of routing security that incorporates the ordinarily overlooked variations in trust that users have for different parts of the network. We focus on anonymous communication, and in particular onion routing, although we expect the approach to apply more broadly.

This paper provides two main contributions. First, we present a novel model to consider the various security concerns for route selection in anonymity networks when users vary their trust over parts of the network. Second, to show the usefulness of our model, we present as an example a new algorithm to select paths in onion routing. We analyze its effectiveness against deanonymization and other information leaks, and particularly how it fares in our model versus existing algorithms, which do not consider trust. In contrast to those, we find that our trust-based routing strategy can protect anonymity against an adversary capable of attacking a significant fraction of the network.

Categories and Subject Descriptors

C.2.2 [**Networks**]: Network Protocols; C.2.0 [**Networks**]: General—*Security and protection*; C.4 [**Performance**]: Modeling techniques

General Terms

Security, Theory

Keywords

anonymous communication, onion routing, privacy, trust

1. INTRODUCTION

Existing anonymous communication theory and system design are generally based on the unrealistic assumption that both adversaries and vulnerability to their attacks are uniformly distributed throughout the communications infrastructure and that a larger network should better protect anonymity. But then if an adversary can control a significant fraction of the network, scaling the network to even tens or hundreds of thousands of nodes will not necessarily improve anonymity. This paper presents a model for routing traffic on an anonymity network where different users trust some parts of the network more than others, potentially allowing users to protect themselves even if large fractions of the network are compromised. We consider route selection for onion routing that makes use of that nonuniform trust and also protects despite an adversary's awareness of it.

While there have been many proposals for anonymous communication protocols [3, 7, 20, 40, 41], onion routing [26] is probably the most dominant. In particular, it enjoys a widely-deployed implementation in the Tor system [17, 49]. As of April 2011, the Tor network's roughly 2500 volunteer routers are used every day by several hundreds of thousands of users to carry about seventy terabytes of traffic [50]. Thus, though our model can apply to many protocols, we focus on onion routing for our examples to illustrate that the theory we introduce can be applied to real systems.

Onion routing, like most anonymous communication paradigms, derives its security from the (traditionally uniform) diffusion of trust throughout the system. In onion routing, users create a cryptographic circuit over a randomly chosen path and then use it to communicate bidirectionally with a destination. Onion routers are supposed to be run by distinct non-colluding entities, but enforcing non-collusion can be difficult. The routers of the Tor network, for example, are operated by volunteers whose identities and intentions are unverified. This choice has provided the network with the diversity and flexibility that has helped it grow to its current scale [16, 18]. But the number of routers that can be added by one entity is limited only by the number of IP addresses he can obtain.

This same general dependence on diffusion of trust applies to most anonymous communication schemes—both deployed anonymity networks, such as Mixmaster [35] and Mixminion [12], and those that have seen more theoretical consideration than actual use, such as the various treatments of Dining Cryptographers [2, 25, 27]. Most of the related research assumes that individual users can do little to learn which nodes are likely to be compromised. But onion routing was originally devised by the U.S. Naval Research Laboratory specifically to target an environment where large organizations or companies could use a network alongside ordinary citizens. What if the user is from an organization that *does* have the resources to investigate nodes, to operate

*This work was primarily done while at the Department of Computer Science at The University of Texas at Austin.

CCS'11, October 17–21, 2011, Chicago, Illinois, USA.

its own nodes, or to otherwise ensure the security of nodes against a particular adversary? Such an organization might run its own private network, in which it controls or vets all the nodes, and just use that. Even if such a private network hides which of its users is responsible for which connection, all traffic exiting it will be known to come from the organization running the network. Alternatively, the organization could run a subnet of the public network and preferentially use that subnet for its own traffic. This approach helps to resist first–last correlation (described below), but it exposes the organization to other attacks. Most significantly, to the extent that the organization is likelier than other users to use its own subnet, all the traffic carried on the subnet will be linked to the organization and therefore to some degree deanonymized. Even if the organization tries to keep its trust and use of its subnet a secret, usage patterns (as would happen if many users from the subnet make requests linkable to the organization) or inadvertent disclosures (as in unauthorized leaks [43]) could over time reveal its presence. We introduce a framework that can address such concerns.

We review related work in the next section. We set out our assumptions, describe our model for the network and adversaries, and provide corresponding definitions for trust and anonymity in Section 3. In Section 4, we use the model to design and analyze a novel path-selection algorithm for onion routing. We begin in Section 4.1 by considering the anonymity of a single connection. In particular, we use trust to obtain an algorithm that improves the posterior probability that an adversary assigns to a given user as the source of a connection given trust levels and the adversary's observations. We consider the value of this posterior for some typical usage scenarios, and compare it to the posterior probability under other path-selection algorithms. We also consider the effect of errors in assigning trust in these scenarios. Next, in Section 4.2, we examine the implications of making multiple connections over time and modify our path-selection algorithm to improve anonymity in this case.

Throughout the paper we try to keep our work applicable to real-world scenarios while remaining abstract enough to permit useful analysis. We hope our work here will provide a foundation for research in route selection so that ultimately users with large-resource, long-reach adversaries can have the assurances necessary to protect their communications.

2. RELATED WORK

The field of anonymous communication has grown vast. For recent general surveys, see Edman and Yener [22] or Danezis et al. [11]. Here we will focus on work related to our central topic, incorporating node trust into route selection. Two types of prior work are thus particularly relevant: First are papers that analyze the anonymity effects of restricted knowledge of the network by route selectors. Second are papers that also use trust in route selection. We also include a brief discussion of first–last correlation attacks.

The first work to consider general effects of route selection on a less than fully connected graph is Danezis's analysis of mix networks with restricted routes [9]. Route restriction was considered to ensure more traffic per link, but he also observed that if the network was an expander graph with N nodes, after $O(\log N)$ random hops a route will have nearly the same distribution on sources as in a fully connected graph.

Danezis and Clayton introduced "route fingerprinting" attacks that exploit the limited knowledge of the network that users must have for P2P anonymity designs to permit scaling [10]. To avoid such knowledge-based attacks, Tor requires that clients know about all the routers in the network. This choice obviously creates scaling problems, but because onion routing is not a P2P design, the number of clients is orders of magnitude larger than the number of routers. This hybrid approach has mitigated both scaling issues and some of the attacks that can arise from partial knowledge of the network. The current work is a generalization from the zero/one trust that is implied by knowledge or ignorance of network nodes [10] to a more fine-grained distinction of willingness to trust a node with one's traffic.

Using trust to make routing decisions in anonymity networks was first explicitly analyzed in [31]. (Prior suggestions of choosing so-called "entry guard" nodes based on trust did not describe how to make this choice or analyze use of trust [39].) Johnson and Syverson considered an adversary that would try to compromise a given fraction of the network's nodes. They used a notion of trust based on difficulty-of-compromise to examine the optimal strategy to resist the first–last correlation attack, depending on the resources of the adversary, the size of the network, and the distribution of trust on the nodes. They did not consider, as we do, that different users could have different distributions on trust or that different users could be concerned with attack by different adversaries. They also considered only how a user could resist correlation attacks given nonuniform trust in the network. They did not attempt a general analysis of other potential attacks in such a network or routing strategies to resist those attacks. Herein we consider additional attacks where the adversary makes inferences based on node selection rather than just trying to see the source and destination.

A very different notion of trust for anonymity networks concerns path formation that considers behavioral trust, such as performance reputation [15, 19]. Sassone et al. analyzed trust in this sense when an adversary compromises a fixed fraction of the network [42]. Users choose paths according to individual trust algorithms (independent of where the adversary exists). They analyze the probability that a user chooses an adversary node, and given that, the probability the adversary attaches to a user creating a path containing that node.

Onion routing's efficiency and bidirectionality make it fast and functional enough for popular online activities like web browsing and instant messaging, which in turn contributes to its success. But onion routing anonymity protocols are not the only ones that have been used for general public communication. In particular, systems based on passing discrete self-contained messages through "mixes" in a source-routed manner similar to onion routing [5, 6] have been used for public Internet communication via email. But even those that are designed to be practical or have been deployed and widely used [12, 28, 35] add much more latency and overhead compared with onion routing.

The added latency in mix systems is not just inefficiency. High-variance latency helps to protect against several types of attacks on anonymity that onion routing does not resist as well or at all, such as the first–last correlation attack [48] or various others [13, 23, 29, 30, 32, 33, 34, 36], although onion routing is more secure than mixing against some attacks [45, 46].

First–last correlation attacks require the adversary to match the timing pattern of messages coming from the user to the timing of messages going to the destination. This matching can either be done passively [1, 39] by simply using the timing pattern created by the user, or actively [51] by delaying messages to create timing watermarks. Extant defenses against first–last correlation are either ineffective in practice (padding) or impractical in effect (delaying and mixing) or, more typically, both. Simulation and experimentation have confirmed the obvious, that such attacks require trivial resources or analysis to be successful. If research does not uncover an effective and practical counter to first–last correlation, onion routing for low-latency use must simply accept it and strive to minimize its impact. For example, Tor contains no mixing or padding in its design [17].

3. MODEL

We describe a model to give semantics to the notion of trust in the context of anonymous communication. The model we present provides a foundation for using trust in designing and analyzing anonymity protocols, specialized to the particular setting of improving resistance to deanonymization and profiling for onion routing systems. It is part of a model intended to be general enough to reason about trust for various anonymity protocols, and for secure routing goals besides anonymity, such as route provenance. The more general model also describes other adversary goals beyond deanonymization and profiling, whether the anonymity protocols use onion routing or another approach. For example, an adversary may want to discover which of the various possible adversaries specific (classes of) users are trying to avoid, which could help indicate something about the likelihood that a given circuit belongs to a given user based on how well the circuit counters a given adversary. An adversary may also want to discover which network nodes are more trusted with respect to which adversaries. Among other things, this could indicate the resources deployed to protect a particular (class of) user's communication. Our focus in this paper is to provide a model and algorithms for onion routing that show how trust-aware route selection can greatly improve resistance to deanonymization and profiling. In particular, using trust can substantially improve security even when an adversary controls a significant portion of the network. The general model is more completely described in [47].

1. Let V be the set of nodes. And let $V = U \cup R \cup D$, where U is a set of users[1], R is a set of onion routers, and D is a set of destinations.
2. Let $E \subseteq \binom{V}{2}$ be the set of network links between nodes.
3. Let $\mathcal{A}$ be the set of adversaries.
4. Let $A_v \subseteq \mathcal{A}, v \in V$, be the adversaries with respect to which a node v wants privacy.
5. Let $C : 2^{\mathcal{A} \times (V \cup E)} \to [0, 1]$ indicate the probability of a pattern of compromise among the nodes and links: for $c \in 2^{\mathcal{A} \times (V \cup E)}$, if $(a, x) \in c$, then adversary a has compromised x. C satisfies $\sum_{c \in 2^{\mathcal{A} \times (V \cup E)}} C(c) = 1$.
6. Let $C_v : 2^{\mathcal{A} \times (V \cup E)} \to [0, 1], v \in V$, indicate the belief node v has in a pattern of compromise among nodes and links. The C_v satisfy $\sum_{c \in 2^{\mathcal{A} \times (V \cup E)}} C_v(c) = 1$.
7. Let $I_v \in \{0, 1\}^*, v \in V$, be the inputs each node uses when running the protocol.

[1]Note that we say 'user' to refer to the human user and to the client software that creates connections on the user's behalf or sometimes to the computer on which that software runs. This overlap should not cause problems at the level at which we model systems. It should be clear from context which usage is intended if the distinction is important.

A protocol is run by the nodes over the network links in order to reach some collective state. For purposes of privacy, the relevant property of the protocol is the set of models that are consistent with the observations of an adversary during the protocol's execution. An adversary makes observations at the nodes and links he has compromised. A probabilistic protocol yields a distribution on the sets of possible models.

Investigating privacy in this model then becomes analyzing how likely the adversary is to be in a position to make good inferences about the model. Privacy may be quantified, for example, by the number of bits of node input learned for certain by the adversary. Or it could be that there are reasonable prior distributions that we can allow the adversary to put on the models, and privacy loss is measured by the mutual information between the observations and the models.

The model includes multiple adversaries. This is an important choice for modeling trust in anonymous communication, because a diverse set of users with varying goals and beliefs is necessary for the set to provide good anonymity. Part of that diversity occurs in the adversaries of a user. That means that we cannot require that each user relies on other network entities in the same way. Allowing users to use the network in different ways while still considering overall communication anonymity from their combined actions is a central issue for protocol design.

The adversaries themselves operate by controlling parts of the network. This models both that an adversary might provide some nodes and links to the network and that he might compromise some that are provided by others. We could restrict the adversary to controlling nodes alone, because an adversary that controls a link could be simulated for purposes of analysis by splitting any link and connecting the halves with a node controlled by the adversary. However, given that several attacks on anonymous communication protocols involve observing the network connections in particular [21, 37], it is useful to formally allow both types of compromise. Also, we allow different adversaries to compromise the same node at the same time.

Trust itself appears in our model in the distributions C_v. Trust in a node or link is given with respect to a set of adversaries $A \subseteq \mathcal{A}$. The trust of user u in, say, network link $e \in E$, with respect to A can be understood as a distribution over the ways 2^A in which the adversaries in A have compromised e. If, for example, the probability in C_u is high that some member of A has compromised e, then we can say that u has low trust in e. Ideally, from the user's perspective, the user's beliefs would be true, that is, C_u would equal C. Our model incorporates erroneous beliefs, though, because a user's beliefs affect her actions and may hurt anonymity when they differ from the truth.

Our analysis will assume a population $N \subseteq U$ of naïve users who think any router is as likely to be compromised as any other. It is within this population of users that we will hide the identity of a given user of the network. This approach is equivalent to assuming that the adversary can rule out all users other than u and the naïve users as being

the source of a connection. Let $m = |R|$ be the number of routers. Let $n = |N|$ be the number of naïve users.

The naïve users share the same adversary, $A_n = \{a_N\}, n \in N$. Other users each have their own adversary, $A_u = \{a_u\}$. No router or destination has any adversary. The set of adversaries is $\mathcal{A} = a_N + \{a_u\}_{u \in U \setminus N}$.

The naïve users $n \in N$ hold the same beliefs about their adversary a_N. They believe each router is independently and equally likely to be compromised: $c^n_{a_N} = c_N$.

We assume user $u \in U$ believes that adversary $a \in \mathcal{A}$ compromises router $r \in R$ independently with probability $c^u_a(r)$. The *trust* of u in r with respect to a is then $\tau^u_a(r) = 1 - c^u_a(r)$. If clear from the context, we will drop the superscript u and the subscript a. The use of probabilities to represent trust reflects the uncertainty about the presence and power of an adversary when coordinating among many different parties over large networks. This uncertainty is best modeled directly, rather than giving the node too much power by assuming a known adversary or giving the adversary too much power by analyzing the worst case.

Furthermore, we assume that u believes with certainty either that a observes all links from u to the routers and destinations or that a observes none of them. That is, u believes with probability either one or zero that $\{u, v\}$ is compromised for all $v \in R \cup D$. Similarly, we assume that u believes with certainty either that a observes all links from a given destination $d \in D$ to R and U (if these conflict on a link in $U \times D$, the user believes that link is compromised). This models whether or not the user believes that he or his destination uses a hostile ISP. It will also be taken to include the case that the user visits a known hostile destination. If an adversary observes all traffic to and from a given user, we say that he observes the source links, and if he observes all traffic to and from a destination, we say that he observes the destination links. Our model can capture varying trust on the links as well as the routers, and incorporating this would better reflect reality; however, adding diverse link probabilities would complicate the analysis below. So, we restrict ourselves to analysis of varying router trust.

Finally, we assume that u does not believe a compromises users, destinations, or links between routers. As noted above, destination compromise is covered by an adversary observing the destination links. Similarly, the case of observed links between routers is subsumed by the adversary compromising either of the onion routers on that link.

Each user has as input a sequence of destinations $(d_1, d_2, \ldots)$ indicating connections over time. Routers and destinations have no inputs.

Anonymous-Communication Privacy.

We assume that users make connections according to a probabilistic process, and that the adversary uses it as a prior distribution to break privacy. Specifically, we assume that the source and destination of a given connection are independent of connections at other times. We also assume that the user and destination of a connection are independent. We acknowledge that this may not be true in practice—users communicate with different partners, application-layer protocols such as HTTP have temporal patterns of connection establishment, and so on. This assumption simplifies analysis, however, and isolates what the adversary can learn by observing the path.

We assume that the adversary's observations consist of a sequence of *active* links, that is, links carrying messages. We further assume that the adversary can determine (for example, via a correlation attack) when two observations correspond to the same connection.

We consider the privacy of the connections that each user makes (the user inputs) to be the most significant among the components of the model. We thus design our protocols to hide information about the connections, and analyze their privacy in most detail. We perform two types of privacy analysis on the connections. First, we consider the ability of the adversary to infer existence, source, and destination of a given connection, that is, to *deanonymize* the connection. Second, we consider the adversary's knowledge of all user connections over time, that is, his *profile* of each user's activity.

Deanonymization. This kind of analysis is useful when the privacy of a given connection is important, say, because it is particularly sensitive. For this analysis, we assume that the adversary has full knowledge of the model except for the user inputs. The analysis uses the posterior probability of deanonymizing the connection as a privacy metric. We want the probability that the adversary correctly names both the user and the destination of a given connection to be close to what it would be if he had not observed the network.

The correlation attack allows the adversary to infer the source and destination of a connection if he can observe both ends. Therefore, if the adversary can observe traffic from the user (either by compromising the entry router or by observing the user's connection to it) and can also observe traffic from the destination (either because he controls the destination or the last router on the path or observes the traffic between them) then the user has no anonymity. Otherwise, the adversary must use the parts of the connection that he can observe and determine the probability of each user being the source.

Profiling. This analysis is useful to understand the adversary's overall view of private inputs, which might be individually private but highly linked with one another. We use the entropy of user connections as a privacy metric in this case. Learning which connections are related helps the adversary to determine the set of destinations visited by some user. Such a profile, taken as a whole, may itself help identify the user if the adversary has background knowledge about the user's typical communication patterns. The adversary might also try to link connections that have identifying information in their traffic with connections that do not, thereby removing anonymity from those that do not and adding profiling information to both kinds of connections.

4. TRUST IN PATH SELECTION

The addition of trust gives users the ability to select routers that are not likely to be compromised by an adversary that they care about. Specifically, depending on various parameters, users of an onion-routing network who choose paths entirely out of highly trusted routers can sometimes minimize their risk of deanonymization via the correlation attack [31]. However, if other users are concerned about adversaries with a different trust distribution, using only highly trusted routers would lead to different users preferring different routers for their paths—and the choice of routers itself may identify the user. For example, an adversary that controls just the last router on a path observes the destination

and the last two routers, and this information alone could deanonymize the user's connection.

By balancing between these effects, we can avoid deanonymization on a single connection. Users make multiple connections over time, however. If their paths change with every new connection, they run an increasing risk of selecting a path that has many compromised routers. An obvious way to avoid this problem is for each user to choose one fixed path to use for all of her connections.

While this strategy helps avoid deanonymization, choosing a single, static path allows an adversary to more easily link together connections from the same user. If the adversary observes the same set of routers in the same positions in two different circuits, he knows it is likely that they originate from the same user. Of particular concern is a malicious destination, because it always observes the *exit router*, that is, the last, static router. Combining static and random router choices allows us to balance avoiding deanonymization with avoiding profiling.

We analyze the impact of the above issues on path deanonymization and use the results of our analysis to motivate a path-selection algorithm. For the adversary types we analyze, the only nontrivial case will be when the adversary compromises destination links and some routers. Briefly stated, our algorithm for this case is to choose a static "downhill" path that picks each successive node from a pool that increases in size because the acceptable lower bound on node trust diminishes with each hop. Once the static path reaches the trust bottom, so that the pool includes all nodes, two dynamic hops are added to the end of the path. We do not claim that this is an optimal strategy. It does demonstrate how to use our model and analysis to easily do better than choosing nodes ignoring trust or using only the most trusted nodes. The details of how our analysis motivates the algorithm are set out below. Our analysis proceeds in two stages: first, we consider minimizing the chance of deanonymization of just a single connection, then second, we consider adapting to multiple connections to maintain good anonymity while also preventing profiling. We will be considering routing for a given user $u \in U$, and we will describe it with respect to the one adversary of u.

4.1 Path anonymity for a single connection

Suppose a user makes just one connection. She chooses a path for that connection based on her trust values for the routers. The adversary can learn about routers on that path by compromising them, compromising an adjacent router, or by observing source or destination links. He can link together routers as belonging to that circuit using the correlation attack. The adversary's ability to determine both source and destination of the circuit, and thereby deanonymize it, depends on these observations. We would like to choose paths to minimize the probability that he can do so.

The best way to select paths depends on the location and kind of adversary we are facing. There are four possibilities depending on whether the source links are compromised or not and whether the destination links are compromised or not. The cases that the source and destination links are either both unobserved or both observed are trivial. The user in these cases can do no better than directly connecting to the destination. If the adversary just observes source links, then we must try to hide the destination. If the adversary just observes destination links (or is the destination), then we must try to hide the user.

4.1.1 Source links observed

Suppose that the adversary observes the source links. Then the user is anonymous if and only if the adversary doesn't observe the destination. Therefore, we can maximize anonymity by choosing a one-hop path that maximizes $\tau(r_1)$, which is achieved by selecting a most-trusted router.

4.1.2 Destination links observed

Now suppose that the adversary observes the destination links. In this case, the adversary is able to learn about the final router, any router he has compromised, and any router adjacent to a compromised router. The adversary can determine that these routers are on the same path and what position they are in by using the correlation attack. Assuming that the adversary knows the user's trust values and the algorithm that the user uses to choose paths, then the adversary can use his observation to infer a distribution on the source of the circuit. We would like to minimize the probability that he assigns to the correct user.

We analyze this probability for a given user u with respect to the population of naïve users (that is, those users with no trust values or identical trust values).

Let P be a random ℓ-hop connection through the onion routing network, with P_i the i^{th} router. Let S be the source (user) and Δ the destination. Let p be a connection made by user u. Let $A_R \subseteq R \cup D$ be the routers and destinations compromised by the adversary. Let the path positions observed by the adversary be $O = \{i : p_i \in A_R \vee p_{i-1} \in A_R \vee p_{i+1} \in A_R \vee (i = \ell \wedge \Delta \in A_R)\}$. Let q_1 be the probability of u making a connection consistent with the routers in p observed and not observed by A_R:

$$q_1 = \left(\frac{1}{n+1}\right) \prod_{i \in O} Pr[P_i = p_i | S = u] \cdot \prod_{i \notin O} \sum_{r \in R \setminus A_R} Pr[P_i = r | S = u].$$

Let q_2 be the probability that a naïve user other than u makes a connection consistent with A_R and p:

$$q_2 = \begin{cases} 0 & \text{if } p_1 \in A_R \\ \left(\frac{n}{n+1}\right) m^{-\ell} |R \setminus A_R|^{|\{i \notin O\}|} & \text{otherwise} \end{cases}$$

These expressions follow from the facts that each user is equally likely to be the source of a given connection and that naïve users choose routers uniformly at random.

Finally, let Y be the conditional probability that the source of a connection is u:

$$Y(A_R, p) = \frac{q_1}{q_1 + q_2}.$$

Y depends on the routers and destinations observed by the adversary and the probability distribution with which u selects a path. The routers in A depend on the trust values of the routers, and the destinations it observes are fixed. Y therefore is probabilistic with a distribution that depends on the path distribution of u and the trust values. The trust values are given, but we can choose the path distribution of u to optimize the distribution of Y.

There are several plausible criteria on the distribution of Y to use when optimizing the path distribution: the expecta-

tion $E[Y]$, a weighted expectation $E[f(Y)]$ for some weight function f, the probability $Pr[Y \geq \alpha]$ for some $\alpha \in (0, 1]$, and so on. Such criteria might lead to different optimal path distributions, because distributions of Y are not necessarily comparable (one may not stochastically dominate the other). We choose the expected value of Y as our criterion.

4.1.3 A Downhill Algorithm

Given the above framework, how can the users utilize trust to better protect their connections? As noted, more trust means lower chance of compromise, hence lower chance of observing the user, adjacent routers in the path, or the destination. On the other hand, a more trusted router is more likely to be associated with the user by someone who knows what adversary the user is trying to avoid. This suggests a routing algorithm in which the user chooses routers from sets with a decreasing minimum trust threshold (or, equivalently, increasing maximum risk of compromise). As these sets increase in size, they are more likely to contain a compromised router, but what that compromised router will see is also less identifying to the user.

Let ℓ be the length of the path. Let the acceptable level of risk in the i^{th} hop be $\rho : [\ell] \rightarrow [0, 1]$ such that $\rho(i) < \rho(i+1)$. The set of routers at the i^{th} level is the "trust set" $T_i = \{r \in R : c(r) \leq \rho(i)\}$. The user chooses the i^{th} hop independently and uniformly at random from T_i.

We can assume that the final trust set T_ℓ includes all of the routers R. The destination links are observed in this case, so we can't give the adversary any more observations by including all of R in a final trust set. And doing so may prevent the adversary from observing a trust set that is smaller than R and thus more associated with the user.

We want to set the parameters ℓ and $\rho(i)$ to minimize the expected posterior probability $E[Y]$. The straightforward algorithm for minimizing the expected value simply iterates over all possible path lengths and ways to set the trust thresholds at each path length. Let $\lambda \leq m$ be the maximum allowed path length. In practice, we only want to consider path lengths up to the point at which the latency becomes too high. The Tor network, for example, uses path lengths of 3. Let ν be the number of distinct trust values among the routers. The number of iterations is then $O(\lambda\nu^\lambda)$.

For each iteration, expected value is determined by calculating the posterior probability the adversary assigns to u for each possible set of compromised routers and each path:

$$E[Y] = \sum_{A_R \subseteq R} \prod_{a \in A_R} c(a) \prod_{a \notin A_R} (1 - c(a)) \sum_{p \in R^\ell} \prod_{i=1}^{\ell} Pr[P_i = p_i | S = u] Y(A_R + d, p).$$

Calculating the expectation from this expression involves summing over all 2^m possible adversary subsets, which takes far too long for any reasonably-sized anonymity network. However, in practice we do not expect the user to be able to make more than a handful of meaningful distinctions between routers on the basis of trust. If the number ν of distinct trust values is small, we can speed up the computation of the expectation by using the fact that our path distribution chooses all routers with the same trust value with equal probability.

4.1.4 Analyzing The Downhill Algorithm

We have calculated the optimal thresholds and the resulting expected posteriors for several plausible situations using a user population of $n = 1000$. The results appear in Table 1. They are given next to the anonymity of two other path algorithms for comparison: *i*) the user chooses each hop uniformly at random from the most-trusted nodes only and *ii*) the user chooses each hop uniformly at random from all routers. We also compare the results to a lower bound on $E[Y]$ of $c_{\min} = \min_r c(r)$. (The first node is compromised with probability at least $c_{\min}$, and $Y = 1$ in this case.) The situations we consider involve three different trust values, so we consider path lengths up to three. In Tables 1(b) and 1(c), the optimal thresholds skip one possible trust value and only use a two-hop path.

Table 1: Examples of optimal thresholds

(a) Small trusted and untrusted sets, for example when the user has information about a few good routers and a few bad routers, and has little information for the rest.

# Routers	5	1000	10
Prob. of compromise	0.01	0.1	0.9
Optimal thresholds	0.01	0.1	0.9

	Downhill	Trusted	Random	Lower bnd.
$E[Y]$	0.0274	0.2519	0.1088	0.01

(b) Small trusted, medium semi-trusted, large untrusted sets, for example when the adversary is strong, but the user and her friends run some routers.

# Routers	5	50	1000
Prob. of compromise	.001	0.05	0.5
Optimal thresholds	0.05	0.5	

	Downhill	Trusted	Random	Lower bnd.
E[Y]	0.0550	0.1751	0.4763	0.001

(c) Equally large trusted, semi-trusted, and untrusted sets, for example when the user assigns trust based on geographic regions.

# Routers	350	350	350
Prob. of compromise	0.1	0.5	0.9
Optimal thresholds	0.1	0.9	

	Downhill	Trusted	Random	Lower bnd.
E[Y]	0.1021	0.1027	0.5000	0.1

The table shows that using trust levels improves anonymity in each case against random route selection by factors of at least 4.0 and as much as 8.6. Similarly, we see improvements in each against the trusted-router strategy, from just a slight increase in the third situation when there are relatively many trusted routers to over a factor 3.1 improvement in the second situation when there are many untrusted routers. We can also see that in the first and third situations we achieve anonymity on the order of the best possible. Interestingly, we notice that in the first situation, using highly-trusted routers exclusively is worse than randomly choosing routers, but only because there are few highly-trusted routers. Using the downhill algorithm avoids that problem.

Figure 1 examines the effect of varying some of the trust values in the situation of Table 1(a). It shows the effect of

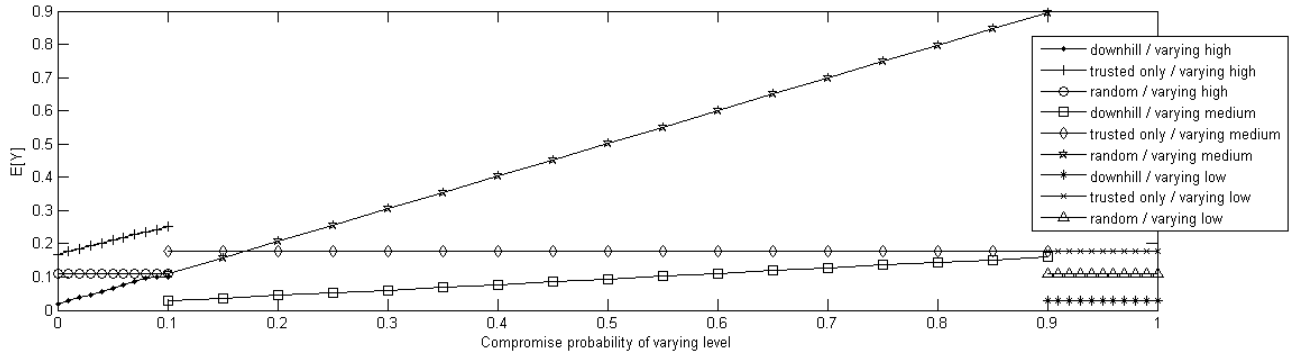

Figure 1: Anonymity when varying trust values of Table 1(a).

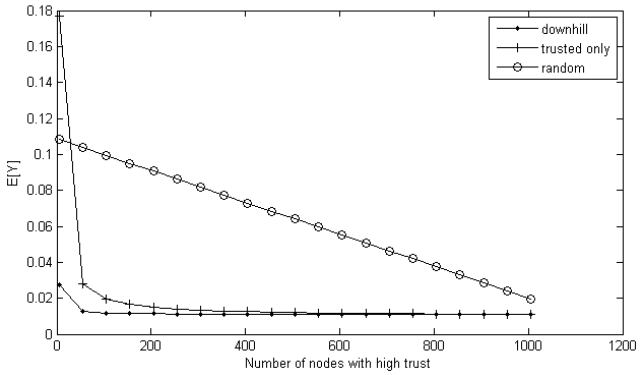

Figure 2: Anonymity when increasing number of high-trust nodes and decreasing number of medium-trust nodes in Table 1(a).

varying just high trust values, keeping the others at their original value, and the effects of the same process with the medium and low trust values. We can see that the change in the anonymity $E[Y]$ is roughly linear in the change in the trust values, and that the rate of the change increases as the trust change affects hops closer to the source. Furthermore, we see that choosing only trusted routers performs badly when the largest trust value isn't high, random selection performs badly when the average trust value isn't high, and the downhill-path algorithm always performs better than both and often much better.

Figure 2 examines the effect of trading off the number of high-trust and medium-trust nodes in the situation of Table 1(a). That is, it shows the variation in anonymity for that situation when there are x high-trust nodes and $1005 - x$ medium-trust nodes. The graph shows that using the downhill-trust or trusted-only algorithms quickly benefit from having larger numbers of high-trust nodes. This is because a selection of these is likely to be observed but not compromised. In contrast, random selection benefits roughly linearly in the number of routers shifted from medium to high trust.

Though these examples show very positive results for plausible scenarios, they are merely illustrative examples. There is no guarantee that they are representative of improvements when using trust values in deployed systems in actual use.

But there is a more immediate concern. In the above calculations, we have assumed that the trust value assigned to a router by the user reflects the correct a priori probability of compromise of that router by the relevant adversary. What if the user was not correct in her assignment of trust?

4.1.5 Correctness and Accuracy of Trust Assignments

To assign trust values, the user must rely on some outside knowledge. This external information might include knowledge of the organizations or individuals who run routers—including both knowledge of their technical competence and the likelihood that those running a given router would intend to attack the user. Trust values might also be affected by computing platforms on which a router is running, geopolitical information about the router, knowledge about the hosting facility where a router might be housed or the service provider(s) for its access to the underlying communications network, and many other factors.

The process of assigning trust values is clearly uncertain, and we cannot expect the user to correctly assign values to all routers. Therefore, we consider the effect of errors in the believed trust values on the user's anonymity. First, we derive a bound on the effect that error in the trust value for a single router has on our anonymity metric, $E[Y]$. Second, we calculate the effect of a couple of types of errors in a specific scenario.

Let $r \in R$ be some router with an error of ϵ in its assumed trust value. Let $E_\epsilon[Y]$ be the expected posterior probability when the probability that r is compromised is $c(r) + \epsilon$. Let $S_i = T_i \setminus T_{i+1}$. Let k_1 be the first path position non-adjacent to the user for which r can be chosen, that is, for $r \in S_1$, let $k_1 = 2$, otherwise let k_1 be such that $r \in S_{k_1}$. Let k_2 be such that $r \in S_{k_2}$. Let μ_i be expected number of uncompromised routers in S_i given that r is uncompromised. Let $\mu_{\min} = \min_{1 \le i \le \ell} \mu_i$. Let P be a random path chosen by u according to the downhill algorithm. Let the probability that r is chosen i times in P be

$$\pi_r(i) = Pr[|\{j : P_j = r\}| = i] \tag{1}$$

$$\le \binom{\ell}{i} \prod_{j=k_1}^{k_1+i-1} 1/|T_j| \prod_{j=k_1+i}^{\ell} (1 - 1/|T_j|). \tag{2}$$

Let the ratio of the probability of u choosing a given router $s \in T_i$ at the ith step to the probability of a given naïve user doing the same be $\alpha_i = m/|T_i|$.

To express our bound succinctly, further let

$$a_i = \prod_{j=k_1-1}^{\min(k_1+3i-2,\ell)} \alpha_j,$$

$$b = \min(2\ell e^{-\mu_{\min}^{1/2}/4}, 1), \text{ and}$$

$$c_i = (1 + O(1/\mu_{\min}))^{\ell-i+1}(1 + O(\mu_{\min}^{-1/4}))^{\min(3i,\ell)}.$$

a_i bounds the relative increase in posterior probability gained from observing additional path positions. We use the Chernoff bound to obtain the bound b on the probability that, for all j, the number of uncompromised routers in S_j is within a factor $1 \pm \mu_{min}^{-1/4}$ of μ_j. c_i represents a bound on the relative posterior increase obtained from losing some unobserved positions and increasing the contribution of the retained unobserved positions. c_i is given explicitly in the proof, and its hidden constants are not large.

Then we can bound the effect of the error in r's trust value as follows:

THEOREM 1. *If* $\mu_{\min} \geq 1$, *then*

$$E_\epsilon[Y] - E[Y] \leq \epsilon\Bigg(Pr[P_1 = r] + (1 - Pr[P_1 = r])\cdot \left(b + (1-b)\sum_{i=0}^{\ell} \pi_r(i)(1 - 1/(c_i a_i))\right)\Bigg).$$

We omit the proof of the theorem for space.

We can see from Theorem 1 that the effect of the trust error is bounded by ϵ. Next, suppose that the expected number of uncompromised routers μ_i in each S_i is large. Suppose also that $r \notin T_1$. Then, the bound provided by the theorem approaches

$$\epsilon\left(\sum_{i=0}^{\ell} \pi_r(i)(1 - 1/a_i)\right).$$

This expression shows that the change in anonymity is determined by how often r is likely to be chosen in the path and how incriminating the observed positions are. Indeed, this expression goes to zero as the probability of choosing r in P goes to zero or as the values α_j of the observations go to one. These events happen when, for example, the smallest trust set containing r (i.e. T_{k_2}) grows, by Inequality 2 and the definition of the α_j.

We examine the concrete effect of trust errors by extending the scenario given in Table 1(a) to include errors. Figure 3 shows the anonymity when the user is incorrect about the trust level of a fraction of the nodes. Specifically, for a fraction x, x of the believed high-trust nodes have medium trust, $x/2$ of the believed medium-trust nodes have high trust, $x/2$ of the believed medium nodes have low trust, and x of the low-trust nodes have medium trust. The figure shows that using trust may actually be worse than choosing randomly if there are significant errors in the user's trust beliefs. In particular, as Theorem 1 describes, performance is particularly sensitive to errors in the most-trusted routers. Thus, even if the average trust values are lower than believed, as in Figure 3, the actual anonymity may be lower than believed if using trust.

Figure 4 also shows the effect of trust errors in the scenario of Table 1(a). The error it shows is an incorrect belief in the middle trust value. The anonymity is compared to that of the downhill algorithm using the correct trust values. We see that they are identical until the middle trust value reaches about .5. This is because the three-hop path is optimal in both cases until then, at which point it becomes optimal to use two hops. This illustrates that trust errors that leave the ordering of nodes by trust roughly the same may not change the optimality of the selected sequence of trust sets.

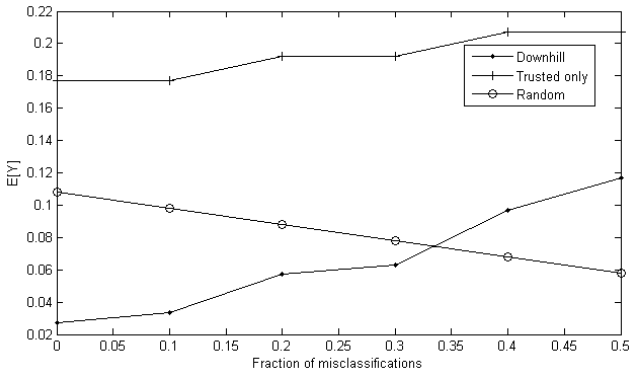

Figure 3: Anonymity when a fraction of nodes of Table 1(a) have incorrect trust values.

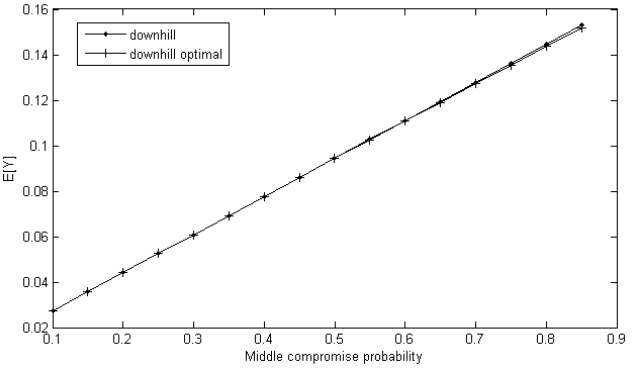

Figure 4: Anonymity from varying trust values of Table 1(a).

4.2 Path selection for multiple connections

The path-selection algorithm described is designed to protect the anonymity of one connection. However, users make multiple connections over time, and we want to maximize the probability that all of them have good anonymity.

If we were to simply use the given algorithm to choose a different path for every connection, users would be increasingly likely to have poor anonymity on at least one of their connections: each new connection would be another chance to select a compromised router. We want to maximize the probability that no connection has poor anonymity. This requirement would suggest that each user should maintain the same path across different connections—similar to the use of guard nodes in Tor suggested by Øverlier and Syverson [39].

However, doing so would make it easier for the adversary to link together different connections as coming from the

same user. Suppose, for example, that the adversary observes the destination links and that users make repeated connections to the destination. The adversary would see multiple connections coming from the same router and can infer that they're more likely to come from the same user.

4.2.1 Path selection against different adversaries

If the adversary observes the source links, then he can already link connections together as belonging to the observed user. Therefore, in this case, we extend the path-selection algorithm from one connection to multiple connections by using the same path for all connections.

The difficult case is again when source links are unobserved, but the adversary observes the links of some of the destinations. For connections that are to unobserved destinations, the user should simply bypass the network entirely. For connections that are to observed destinations, we adapt the one-connection algorithm by using both static and dynamic components. First, as given in that algorithm, the user uses a decreasing sequence of trust thresholds to choose a path of length ℓ. At each hop, a router is chosen uniformly at random from all routers with trust value above the threshold at that hop. This path is *static*—it is chosen once at the start, and then the user uses it for all connections. Second, two routers are chosen uniformly at random from R and used as the $(\ell+1)^{st}$ and $(\ell+2)^{nd}$ hops. These hops are *dynamic*—a new random selection of these two hops is made for each new connection.

Combining static and dynamic hops in this way helps maintain anonymity over multiple connections while preventing them from being easily linked. The static portion of the path protects the source identity over all connections by preventing the source and her most-trusted routers from being observed even once. The last hop is dynamic so that the adversary observing destination links cannot use a static router to link together repeated connections from the same user. The $(\ell+1)^{st}$ hop is dynamic because the last hop is likely to be compromised on a fraction of connections equal to the fraction of routers that are compromised by the adversary. If this hop were static, the adversary could use it to link together destinations it observes from the last hop.

Of course, the $(\ell+1)^{st}$ router is also likely to be compromised on a fraction of the connections. When those connections are to destinations for which the links are observed, then they can be linked because of the static ℓ^{th} hop. However, multiple connections might not be only to such destinations. Due to uncertainty about the destination links or just for simplicity, users could use the downhill-trust algorithm to destinations with unobserved links. By using two dynamic routers as the final hops, linking unobserved destinations requires that both dynamic routers be compromised. Note that adding additional dynamic routers at the end provides no benefit, as the adversary can perform a correlation attack using only the first and last dynamic routers in order for the destinations to be observed and linked.

4.2.2 Analyzing path selection

In order to rigorously analyze the effectiveness of using static and dynamic routers together, we consider the problem of linking more precisely. The use of static components in the path means that if the adversary observes two different connections using the same static hops, they are more likely to belong to the same user.

Therefore, instead of looking only at his knowledge of a given connection, we must examine the adversary's overall view of which connections occurred. A user's private information consists of the sequence of connections that user makes over a given time period, where each connection consists of a user, a destination, and a start time. We examine user privacy over multiple connections by looking at the adversary's posterior distribution on the number of connections at the user's connection-start times and the sources and destinations of those connections. We use the entropy [14, 44] of this distribution as our metric of uncertainty. We will consider how this entropy is affected by adding dynamic routers at the end of the paths of some users.

Let $A_R \subseteq R$ be the routers compromised by the adversary. Let T be the set of start times of the connections for which u is the source. Let C^T be a random binary vector indicating the presence of a connection starting at the times in T. Let S^T be a random sequence of users indicating the sources for connections starting at times in T. Let D^T be a random sequence of destinations indicating the destinations for connections starting at times in T. Finally, let O_s indicate the adversary's observations when users only create and use the static part of the path, let O_{d_1} indicate the observations made by the adversary when users add one dynamic hop, and let O_{d_2} indicate the adversary's observations when users add a second dynamic hop after the first.

We are interested in how the entropy of the posterior distribution over connections given an adversary's observations changes when using dynamic hops. That is, we consider how $H(C^T, S^T, D^T|O_S)$, $H(C^T, S^T, D^T|O_{d_1})$, and $H(C^T, S^T, D^T|O_{d_2})$ compare. Using the chain rule of entropy [8] and the independence of S^T and D^T, we can express the entropy of the posterior as

$$H(C^T, S^T, D^T|O) = H(C^T|O) + H(S^T|C^T, O) + H(D^T|C^T, O), \quad (3)$$

where O is O_{d_1}, O_{d_2}, or O_s.

We first show that, assuming a user only makes connections over the anonymity network to destinations with observed links, then adding one dynamic hop to the static path can only increase the entropy over her connections.

THEOREM 2.

$$H(C^T, S^T, D^T|O_{d_1}) \geq H(C^T, S^T, D^T|O_s).$$

PROOF. The entropy $H(C^T|O = o)$ of the existence of connections at times in T does not change due to dynamic hops because the connections are always observed at the destination links.

The entropy $H(S^T|C^T, O = o)$ of connection sources can only change when a final static hop of a connection goes from being observed to unobserved. The final static router can become unobserved if the final static router is uncompromised and the penultimate static router is uncompromised. Then it becomes unobserved when the dynamic hop is uncompromised. To understand how the entropy can change, suppose the final static hop goes from being unobserved to being observed by removing the dynamic hop. Consider any value s^T of S^T and some set of observations o. s^T and o together imply a set of path selections for users in s^T. The conditional probability $Pr[S^T = s^T|O = o]$ is proportional

to the probability that each user made the implied path selections. Removing a dynamic router from the end of a given connection can change this probability in two ways. First, the probability can decrease by a factor $1/m$, as the observation of the final static router implies that the source of the connection in s^T chose it in her path, and we can assume that the final static hop is chosen randomly from R. Second, it can send it to zero, if the source s^T assigned to the given connection is also assigned to another connection that is observed with a different final static router. Thus, the entropy can only decrease when the final static hop goes from being unobserved to being observed. This implies that the entropy can only increase when the final static hop switches from being observed to being unobserved.

The entropy $H(D^T|C^T, O = o)$ of connection destinations does not change, because all destinations of the user are assumed to be observed by the adversary.

Because no term of Equation 3 can decrease, the overall connection entropy $H(C^T, S^T, D^T|O = o)$ cannot decrease either. □

Next, we show that, again assuming a user only makes anonymous connections to observed destinations, using two dynamic hops has the same entropy as using one dynamic hop.

THEOREM 3.

$$H(C^T, S^T, D^T|O_{d_2}) = H(C^T, S^T, D^T|O_{d_1}).$$

PROOF. The proof of Theorem 2 shows that the first dynamic hop cannot change the entropy $H(C^T|O)$ of the connections or the entropy $H(D^T|C^T, O)$ of the destinations. Adding a second hop cannot change these for the same reason. In addition, the second hop cannot change the entropy $H(S^T|C^T, O)$ of the sources because it is only adjacent to the first dynamic hop, which is chosen at random by all users. Therefore, by Equation 3, adding a second hop cannot change the entropy $H(C^T, S^T, D^T|O_{d_1})$. □

Theorems 2 and 3 only show that adding dynamic hops doesn't decrease the connection entropy. This is not a particularly strong justification of their use. However, in general, this is the strongest claim we can make. Adding dynamic hops can increase the entropy by little or none if *i*) the adversary controls most of the network and thus most of the dynamic routers, *ii*) the adversary has compromised the user's initial, ultimate or penultimate static hops, or *iii*) each user has a unique pattern of observed hops not including the last hop.

5. CONCLUSION AND FUTURE WORK

The existing anonymous communication definitions and system models typically assume that all nodes in the network are the same, and that any part of the system is as likely to threaten security as any other. In this paper we have set out the first network and adversary model for anonymous communication that accounts for the diversity of trust that different users may have in elements of the network. The presented model is a specification for onion routing of a more general model we have developed to reason about various approaches to routing security [47]. We identified two important classes of privacy attacks in this model and presented an example of a routing algorithm motivated by resistance to them. Analysis of this algorithm in our model shows that it significantly improves the anonymity of onion routing, especially when an adversary can compromise a large fraction of the network.

An adversary that learns the trust placed in specific routers may learn something about the resources a user (or her organization) has applied to protect her communications. And, trust information must first be learned to be used in deanonymization as analyzed in Section 4.1. Given available space, we leave any discussion of how an adversary might learn trust values to future work. Similarly we do not discuss an adversary-learning adversary for either the usage scenario and algorithm we have described or for other cases. (For example, if Alice is a so-called "road warrior" travelling on behalf of her employer, and she wishes to log in from her hotel to her workstation back at her office, starting her circuits at highly trusted nodes would reveal something about who she is trying to hide from. Such a setting requires a complementary uphill-trust algorithm, although the complement is not simply a reverse of the downhill algorithm. There are other subtleties, such as the different need for dynamic hops.)

As in onion-routing networks, trust can play a role in mix networks too by helping to avoid compromised routers. But the adversary and communication assumptions are somewhat different, leading to different strategies. Our general model encompasses both of these types of anonymous communication as well as others; however, in this paper we limit discussion to onion-routing networks. We would like to investigate the impact of trust on other protocols to provide secure routes. The following are some of the other issues we intend to explore in future work: What impact might more user classes trying to avoid distinct nonuniform adversaries have on each other? How robust are our results when a fraction of users defect from the strategy that is optimal for a given non-naïve user, and what incentive mechanisms can induce them to cooperate? Note that users with flat trust distributions will choose ℓ=1 plus two dynamic hops—which is exactly the path-selection algorithm that Tor uses today [39]. The adversaries could employ other methods of attack, such as congestion attacks [23, 36], DoS attacks [4], changing the network topology, and manipulating the trust values. In addition to routers, we would like to investigate the effect of different trust values among links, to account for real-world Internet routing issues [21, 24]. We would like to investigate a *roving* adversary that tries to compromise different sets over time [38, 48]. The joint distribution of the events that adversaries compromise nodes could be arbitrary, instead of assuming independence of compromise between nodes. We could give the attacker a budget and assign costs to attempting compromise on a node. Users might have multiple adversaries. Every user could set a cost for every adversary of losing privacy to that adversary. Thus we could model the case where Alice doesn't want to be exposed by either Eve or Mallory, but would prefer one to the other.

Though just a simple example, we have shown that our new routing algorithm has significant security impact in plausible usage scenarios. Our model incorporates trust-based routing, a novel aspect of both anonymous communication in particular and secure communication in general. We hope that other researchers will see the potential of our

approach and take up the above questions or be inspired to explore our model through questions of their own.

6. ACKNOWLEDGMENTS

The authors would like to thank Nick Hopper for much guidance in improving this paper. We would also like to thank George Danezis, Karsten Loesing, and the anonymous reviewers for helpful comments on drafts of this paper. This work supported by ONR, NSF, and DARPA.

7. REFERENCES

[1] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker. Low-resource routing attacks against Tor. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2007)*, Washington, DC, USA, October 2007.

[2] R. Beauxis and C. Palamidessi. Probabilistic and nondeterministic aspects of anonymity. *Theoretical Computer Science*, 410(41):4006–4025, 2009.

[3] A. Beimel and S. Dolev. Buses for anonymous message delivery. *Journal of Cryptology*, 16(1):25–39, 2003.

[4] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz. Denial of service or denial of security? How attacks on reliability can compromise anonymity. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS)*, October 2007.

[5] J. Camenisch and A. Lysyanskaya. A formal treatment of onion routing. In V. Shoup, editor, *Advances in Cryptology – CRYPTO 2005: 25th Annual International Cryptology Conference*, pages 169–187. Springer-Verlag, LNCS 3621, August 2005.

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

[7] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology: The Journal of the International Association for Cryptologic Research*, 1(1):65–75, 1988.

[8] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[9] G. Danezis. Mix-networks with restricted routes. In R. Dingledine, editor, *Privacy Enhancing Technologies: Third International Workshop, PET 2003*, pages 1–17. Springer-Verlag, LNCS 2760, 2003.

[10] G. Danezis and R. Clayton. Route fingerprinting in anonymous communications. In *Sixth IEEE International Conference on Peer-to-Peer Computing, P2P 2006*, pages 69–72. IEEE Computer Society Press, 2006.

[11] G. Danezis, C. Diaz, and P. Syverson. Anonymous communication. In B. Rosenberg, editor, *Handbook of Financial Cryptography*. CRC Press, 2010.

[12] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, 2003.

[13] G. Danezis and A. Serjantov. Statistical disclosure or intersection attacks on anonymity systems. In *Proceedings of 6th Information Hiding Workshop (IH 2004)*, pages 293–308, 2004.

[14] C. Díaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In R. Dingledine and P. Syverson, editors, *Privacy Enhancing Technologies, Second International Workshop, PET 2002, Revised Papers*, pages 54–68. Springer-Verlag, LNCS 2482, 2003.

[15] R. Dingledine, M. J. Freedman, D. Hopwood, and D. Molnar. A reputation system to increase MIX-net reliability. In I. S. Moskowitz, editor, *Information Hiding: 4th International Workshop, IH 2001*, pages 126–141, Pittsburgh, PA, USA, April 2001. Springer-Verlag, LNCS 2137.

[16] R. Dingledine and N. Mathewson. Anonymity loves company: Usability and the network effect. In R. Anderson, editor, *Fifth Workshop on the Economics of Information Security (WEIS 2006)*, June 2006.

[17] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, pages 303–319. USENIX Association, August 2004.

[18] R. Dingledine, N. Mathewson, and P. Syverson. Deploying low-latency anonymity: Design challenges and social factors. *IEEE Security & Privacy*, 5(5):83–87, September/October 2007.

[19] R. Dingledine and P. Syverson. Reliable MIX cascade networks through reputation. In M. Blaze, editor, *Financial Cryptography, 6th International Conference, FC 2002*, pages 253–268. Springer-Verlag, LNCS 2357, 2003.

[20] S. Dolev and R. Ostrovsky. Xor-trees for efficient anonymous multicast and reception. *ACM Transactions on Information and System Security*, 3(2):63–84, 2000.

[21] M. Edman and P. Syverson. AS-awareness in Tor path selection. In S. Jha, A. D. Keromytis, and H. Chen, editors, *CCS'09: Proceedings of the 16th ACM Conference on Computer and Communications Security*, pages 380–389. ACM Press, 2009.

[22] M. Edman and B. Yener. On anonymity in an electronic society: A survey of anonymous communication systems. *ACM Computing Surveys*, 42(1), 2010.

[23] N. S. Evans, R. Dingledine, and C. Grothoff. A practical congestion attack on Tor using long paths. In *Proceedings of the 18th USENIX Security Symposium*, pages 33–50, Montreal, Canada, August 2009. USENIX Association.

[24] N. Feamster and R. Dingledine. Location diversity in anonymity networks. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2004)*, pages 66–76, 2004.

[25] S. Goel, M. Robson, M. Polte, and E. G. Sirer. Herbivore: A scalable and efficient protocol for anonymous communication. Technical Report 2003-1890, Cornell University, Ithaca, NY, February 2003.

[26] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding routing information. In *Information Hiding: First International Workshop, Proceedings*, pages 137–150. Springer-Verlag, LNCS 1174, 1996.

[27] P. Golle and A. Juels. Dining cryptographers revisited. In *Advances in Cryptology – EUROCRYPT 2004*, pages 456–473, Interlaken, Switzerland, May 2004. Springer-Verlag, LNCS 3027.

[28] C. Gülcü and G. Tsudik. Mixing E-mail with Babel. In *Proceedings of the Network and Distributed Security Symposium (NDSS 1996)*, pages 2–16, 1996.

[29] A. Hintz. Fingerprinting websites using traffic analysis. In R. Dingledine and P. Syverson, editors, *Privacy Enhancing Technologies: Second International Workshop, PET 2002*, pages 171–178, San Francisco, CA, USA, April 2002. Springer-Verlag, LNCS 2482.

[30] N. Hopper, E. Y. Vasserman, and E. Chan-Tin. How much anonymity does network latency leak? *ACM Transactions on Information and System Security*, 13(2):13–28, 2010. February.

[31] A. Johnson and P. Syverson. More anonymous onion routing through trust. In *22nd IEEE Computer Security Foundations Symposium, CSF 2009*, pages 3–12, Port Jefferson, New York, July 2009. IEEE Computer Society.

[32] D. Kesdogan, D. Agrawal, and S. Penz. Limits of anonymity in open environments. In *Proceedings of Information Hiding Workshop (IH 2002)*, 2002.

[33] M. Liberatore and B. N. Levine. Inferring the source of encrypted HTTP connections. In R. N. Wright, S. De Capitani di Vimercati, and V. Shmatikov, editors, *CCS'06: Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 255–263. ACM Press, 2006.

[34] N. Mathewson and R. Dingledine. Practical traffic analysis: Extending and resisting statistical disclosure. In D. Martin and A. Serjantov, editors, *Privacy Enhancing Technologies, 4th International Workshop, PET 2004, Revised Selected Papers*, pages 17–34. Springer Verlag, LNCS 3424, 2005.

[35] U. Möller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster Protocol — Version 2. Draft, 2003.

[36] S. J. Murdoch and G. Danezis. Low-cost traffic analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 183–195, 2005.

[37] S. J. Murdoch and P. Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In N. Borisov and P. Golle, editors, *Proceedings of the Seventh Workshop on Privacy Enhancing Technologies (PET 2007)*, Ottawa, Canada, June 2007. Springer.

[38] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *Proceedings of the Tenth ACM Symposium on Principles of Distributed Computing (PODC '91)*, pages 51–59. ACM Press, 1991.

[39] L. Øverlier and P. Syverson. Locating hidden servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*. IEEE CS, May 2006.

[40] C. Rackoff and D. R. Simon. Cryptographic defense against traffic analysis. In *Proceedings of ACM Symposium on Theory of Computing*, pages 672–681, 1993.

[41] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[42] V. Sassone, S. Hamadou, and M. Yang. Trust in anonymity networks. In P. Gastin and F. Laroussinie, editors, *CONCUR 2010 - Concurrency Theory: 21st International Conference*, pages 48–70. Springer-Verlag, LNCS 6269, 2010.

[43] B. Schneier. Secret German IP addresses leaked. Schneier on Security, `http://www.schneier.com/`, November 2008.

[44] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In R. Dingledine and P. Syverson, editors, *Privacy Enhancing Technologies, Second International Workshop, PET 2002, Revised Papers*, pages 41–53. Springer-Verlag, LNCS 2482, 2003.

[45] P. Syverson. Why I'm not an entropist. In *Seventeenth International Workshop on Security Protocols*. Springer-Verlag, LNCS, 2009. Forthcoming.

[46] P. Syverson. Sleeping dogs lie in a bed of onions but wake when mixed. In *4th Hot Topics in Privacy Enhancing Technologies (HotPETs 2011)*, July 2011.

[47] P. Syverson, A. Johnson, R. Dingledine, and N. Mathewson. Trust-based anonymous communication: Adversary models and routing algorithms. Technical report, University of Texas at Austin, 2011.

[48] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an analysis of onion routing security. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, Proceedings*, pages 96–114. Springer-Verlag, LNCS 2009, July 2001.

[49] The Tor project home page. `https://www.torproject.org/` .

[50] Tor metrics portal. https://metrics.torproject.org/, April 2011.

[51] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao. On flow correlation attacks and countermeasures in mix networks. In *Proceedings of Privacy Enhancing Technologies workshop (PET 2004)*, pages 207–225, 2004.