Inference attacks against trust-based onion routing: Trust degree to the rescue

Peng Zhou*, Xiapu Luo, Rocky K.C. Chang
Department of Computing, The Hong Kong Polytechnic University, Hunghom, Hong Kong

ABSTRACT

Trust-based onion routing enhances anonymity protection by means of constructing onion circuits using trust-based routers. However, attackers who have the knowledge of a priori trust distributions are still capable of largely reducing the anonymity protected by trust-based circuits. The root cause is that these attackers have a high probability to guess the users who initiate trust-based circuits through the routers trusted by few other users (i.e., inference attacks). In this paper, we uncover trust degree, an essential feature of routing anonymity that is effective in defeating inference attacks but has been overlooked in the design of existing trust-based onion routing. We conduct an isolated model based analysis to understand why the trust degree is effective and how it can be used to resist inference attacks. Our major contributions are three-fold. First, we present a model to exclusively reason about inference attacks in trust-based onion routing. This model isolates the anonymity compromised by inference attacks from other attacks (e.g., correlation-like attacks), and hence derives an exclusive design space that reveals trust degree as the key feature against inference attacks. Second, to show the usefulness of our model, we design a new routing algorithm by taking into account of trust degree. Our algorithm can protect anonymity against inference attacks without sacrificing the capability against attackers’ routers. Third, we compare trust-based routing algorithms with and without considering trust degree using real-world social networking datasets. These comparisons present evidence to confirm the effectiveness of trust degree in defeating inference attacks under real-world settings.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

Recently, trust-based onion routing has become a hot research topic (Puttaswamy et al., 2008; Johnson and Syverson, 2009; Danezis et al., 2010; Johnson et al., 2011; Mittal et al., 2013; Zhou et al., 2013), because it can effectively protect anonymity even if a significant fraction of the network is compromised. Onion routing protects anonymity by hiding user identities behind onion circuits. Each onion circuit is comprised of a sequence of layered onion routers. However, since existing onion routing networks such as Tor (Dingledine et al., 2004) do not deploy identity checking mechanisms to verify the identities of onion routers. However, since existing onion routing networks such as Tor (Dingledine et al., 2004) do not deploy identity checking mechanisms to verify the identities of onion routers.
from various correlation-like attacks (Evans et al., 2009; Murdoch and Danezis, 2005; Øverlier and Syverson, 2006; Bauer et al., 2007; Fu and Ling, 2009; Ling et al., 2009; Zhu et al., 2009; Hopper et al., 2010). Using traffic watermarking or traffic analysis techniques, the attackers who control the first and the last routers in users’ onion circuits can correlate these routers in the same circuit, and hence reveal which user visits which destination. To tackle these correlation-like attacks, trust-based onion routing (Johnson and Syverson, 2009; Johnson et al., 2011) has been proposed. By incorporating trust for routing algorithms, trust-based onion circuits can be constructed using trustworthy onion routers and therefore significantly limit correlation-like attacks.

Although trust-based onion routing is proved effective against correlation-like attacks, it induces a new kind of attacks, called inference attacks, due to biased a priori trust distributions (Johnson et al., 2011). As trust-based onion routing cannot be designed to protect anonymity by means of obscurity, the powerful attackers who have the knowledge of a priori trust relationships among users and routers are potentially existing. These attackers have a high probability to guess the initiate user of a trust-based onion circuit if they can observe onion routers in this circuit (i.e., inference attacks). As discussed in Johnson et al. (2011), the inference attack poses a major threat to trust-based onion routing because it can largely reduce the anonymity protected by trust-based onion circuits.

Defeating the inference attack in trust-based onion routing is a very challenging problem, because different users usually have preferences for different sets of onion routers according to trust. The state-of-the-art countermeasures usually make a tradeoff between the capability against inference attacks and the capability against attackers’ routers (e.g., correlation-like attacks are performed based on these routers), and show their effectiveness in terms of overall anonymity (Danezis et al., 2010; Johnson et al., 2011; Mittal et al., 2013; Zhou et al., 2013). These countermeasures cannot differentiate the anonymity compromised by inference attacks from other attacks, hence missing opportunities to disclose key features against inference attacks in the design space. For example, a downhill algorithm is proposed to construct trust-based onion circuits using a decreasing trust threshold among these circuits (Johnson et al., 2011). Trust threshold is an indirect feature that can only mitigate inference attacks by sacrificing the capability against attackers’ routers. Moreover, although onion routing through social networks (Danezis et al., 2010; Mittal et al., 2013) or trust graphs (Zhou et al., 2013) shows better protection of the overall anonymity, the features used to thwart inference attacks cannot be isolated and analyzed independently.

Motivation: To effectively protect anonymity against inference attacks, a fundamental challenge is to discover the key features in the design space. These features can offer distinct advantages for the design of effective and free-of-tradeoff countermeasures, but unfortunately have not been discovered by previous studies.

Methodology: In this paper, we address this fundamental challenge by deriving and analyzing a novel inference attack model. We first model trust-based onion routing as a probabilistic hypergraph, and then isolate the design space for inference attacks by truncating this hypergraph. Based on the analysis of this isolated design space, we uncover trust degree as the key feature against inference attacks. We design a proof of concept routing algorithm by taking advantage of our findings, and confirm the effectiveness of trust degree under real-world settings.

Contributions: We list our main contributions in this paper as follows.

1. We present a novel attack model to isolate the anonymity compromised by inference attacks from other attacks. Compared with generic models that focus on overall anonymity, our model can exclusively reason about inference attacks without being affected by irrelevant features.
2. By analyzing our model, we uncover trust degree as the key feature against inference attacks. Trust degree is very effective in preventing inference attacks, but is overlooked due to the generic model based analysis in prior research.
3. We design a proof of concept routing algorithm by taking into account of trust degree. Our routing algorithm makes no tradeoffs between the capability against inference attacks and the capability against attackers’ routers. It can be embedded into existing trust-based routing algorithms (Johnson and Syverson, 2009; Johnson et al., 2011), or even existing inference attack countermeasures (e.g., the downhill algorithm (Johnson et al., 2011)), to further strengthen the protection of anonymity.
4. We compare trust-based routing algorithm and the downhill algorithm with and without considering trust degree using real-world social networking datasets. The results confirm that trust degree is a very effective feature against inference attacks.

Although we concentrate on the analysis of inference attacks in this paper, we expect the principle of isolating design space for a particular attack could be applied broadly. The isolated attack model is attractive due to its ability of extracting key features. These features are usually very effective against the particular attack, but hard to be discovered by the analysis using generic models.

Paper structure: The remainder of this paper is organized as follows. We first review related work in the literature in Section 2. We then present a novel attack model to isolate inference attacks in Section 3. We reveal trust degree as the key feature against inference attacks and present a new routing algorithm by taking advantage of trust degree in Section 4. We confirm the effectiveness of trust degree using real-world social networking datasets in Section 5. After several discussions in Section 6, we conclude this paper in Section 7.

2. Background and related work

2.1. Correlation-like attacks against onion routing

The onion routing network is one of the most dominated anonymity systems deployed in the Internet today (Dingledine et al., 2004). It protects user anonymity by means of layered encrypted onion circuits (Syverson et al., 1997; Dingledine et al., 2004). These circuits consist of several dynamically
selected onion routers, hence preventing attackers from linking the initiate user and the destination this user visits.

However, the attackers who control a large fraction of the network can pose a serious security threat to onion routing networks. They can employ the routers under their control to actively embed watermarks into onion routing traffic (Fu and Ling, 2009; Ling et al., 2009; Zhu et al., 2009; Luo et al., 2010, 2011) or passively analyze patterns of the traffic (Evans et al., 2009; Murdoch and Danezis, 2005; Øverlier and Syverson, 2006; Bauer et al., 2007; Zhu et al., 2009; Hopper et al., 2010), hence correlating the initiate user and the destination of the same onion circuit. These correlation-like attacks can significantly reduce users’ anonymity protected by onion routing. As an example, if both the first and last routers in a user’s onion circuit are compromised by attackers, this user and the destination this user visits can be linked immediately.

Onion routing networks cannot well protect anonymity against correlation-like attacks, because they cannot effectively verify the trustworthiness of onion routers in the networks (Dingledine et al., 2004). Any people, no matter they are honest humans or attackers and no matter they are network security experts or computer idiots, can act as volunteers to deploy routers in onion routing networks. Onion routing users have no means to check the safety and identity of onion routers, hence probably constructing onion circuits using the routers controlled by attackers.

2.2. Inference attacks against trust-based onion routing

To defeat correlation-like attacks, recent research has proposed trust-based onion routing (Puttaswamy et al., 2008; Johnson and Syverson, 2009; Johnson et al., 2011). Using the trust that users readily have in the owners of onion routers, trust-based onion routing is capable of verifying the identities of onion routers and consequently excluding the routers controlled by attackers with a high confidence. In the literature, Puttaswamy et al. (2008) conduct a pioneer research and propose the use of social links to enhance onion routing. This algorithm restricts onion routing users to select routers from their one- or two-hop neighbors in an online social network. Unlike this pioneer algorithm that only exploits social links, Johnson et al. (Johnson and Syverson, 2009; Johnson et al., 2011) are the first who advocate the onion routing through trust-based circuits. They propose a generic model to reason about trust and derive trust-based onion routing algorithms.

Although trust-based onion routing shows its effectiveness in preventing correlation-like attacks, it reduces the uncertainty of the whole network because a priori trust distributions are usually biased. As a result, the highly capable attackers who knows a priori trust relationships can perform inference attacks to significantly degrade the anonymity protected by trust-based onion routing. Inference attackers have a high probability to guess the initiate user of trust-based circuits if they have the chance to observe onion routers in these circuits. The inference attack attracts considerable attention in recent research (Danezis et al., 2010; Johnson et al., 2011; Mittal et al., 2013; Zhou et al., 2013), because it could be the largest threat in trust-based onion routing.

2.3. Countermeasures against inference attacks

To evade inference attacks, a downhill algorithm as a pioneer solution has been proposed (Johnson et al., 2011). This algorithm enables users to select trust-based routers uniformly at random from sets with a decreasing trust threshold along onion circuits. Although this algorithm targets to prevent inference attack, it does not discover and analyze the key features against this attack. As a result, the downhill algorithm can only mitigate inference attacks at the cost of the capability against attackers’ routers (i.e., a low trust threshold results in a high probability of inducing attackers’ routers to users’ onion circuits).

On the other hand, social network based onion routing has been proposed to implicitly address inference attacks as well (Danezis et al., 2010; Mittal et al., 2013; Zhou et al., 2013). This kind of routing algorithms usually implement random walks on top of social networks to integrate social trust for onion routing. For example, Drac (Danezis et al., 2010) constructs onion circuits by randomly walking over an online social network. However, as Drac performs the conventional random walk, it could be dominated by a few onion routers. To prevent this domination, Pisces (Mittal et al., 2013) proposes a Metropolis-Hastings modified random walk instead. SGor (Zhou et al., 2013), on the other hand, derives global trust from a trust graph to limit inference attacks. Since these social network or trust graph based onion routing algorithms are designed and evaluated in terms of overall anonymity, they cannot conduct in-depth analysis to discover which features are effective against which attacks. Therefore, these routing algorithms result in tradeoff solutions.

In contrast to Danezis et al. (2010), Johnson et al. (2011), Mittal et al. (2013) and Zhou et al. (2013), we seek countermeasures against inference attacks following a very different research principle. We isolate the design space of inference attacks and uncover the key features to design the countermeasures without tradeoffs. Our countermeasures can be embedded into existing routing algorithms and effectively protect anonymity against inference attacks in trust-based onion routing.

3. Inference attack model in trust-based onion routing

In this section, we reason about inference attacks in trust-based onion routing. We first describe the anonymity that onion routing can protect in Section 3.1. We then elaborate on the trust which can be used to prevent attackers’ routers but induces inference attacks in Section 3.2. After listing the attacking capabilities that inference attackers have in Section 3.3, we propose a novel attack model to isolate the anonymity compromised by inference attacks in Section 3.4.

3.1. The anonymity

Onion routing protocols are designed to prevent an attacker from linking users and destinations that the users visit in the Internet. There are two kinds of anonymity that onion routing can protect. One is user anonymity and the other is
destination anonymity. The user anonymity concerns the protection of user identities, while the destination anonymity considers the protection of which destination users visit. Since Johnson et al. (2011) argued in their work that the destination anonymity can be best protected using a single hop onion circuit consisting of the most trusted router, we focus on user anonymity in this paper.

3.2. Trust graph

Following prior research of trust-based onion routing (Puthawamy et al., 2008; Johnson and Syverson, 2009; Danezis et al., 2010; Johnson et al., 2011; Mittal et al., 2013; Zhou et al., 2013), we consider the trust that users have readily assigned to onion router owners in this paper. This notion of trust contains two-fold meanings (Johnson and Syverson, 2009; Johnson et al., 2011). One is the probability that the router’s owner is an attacker itself. A lower probability means a higher level of trust. The other meaning is the difficulty that an honest person’s router can be compromised by attackers. A higher level of difficulty indicates a higher level of trust. Using this notion of trust, the capability of evading attackers’ routers can be measured in terms of the level of trust.

By applying this notion of trust to the whole trust-based onion routing network, we model a priori trust relationships among users and onion routers as a weighted directed trust graph $G$. Let $U$ be the set of users. Let $R$ be the set of onion routers (or the owners of onion routers). We have $G = (U \cup R, U \times R)$, where $U \cup R$ is the set of vertices and $U \times R$ is the set of directed edges in $G$. Each edge $(u, r) \in U \times R$ represents a trust relationship from a user $u \in U$ to an onion router $r \in R$ and is associated with a weight $t(u, r)$ to indicate the level of this trust. Note that, since a person can play the role as a user and a router’s owner at the same time, $G$ is not a bipartite graph (generally $U \cap R \neq \emptyset$).

Since users need outside knowledge to estimate the trustworthiness of onion routers, Johnson and Syverson (2009) and Johnson et al. (2011) argue that users can only have a very coarse level of trust for onion routers. Almost existing research (Puthawamy et al., 2008; Johnson and Syverson, 2009; Danezis et al., 2010; Mittal et al., 2013; Zhou et al., 2013) designs their trust-based onion routing algorithms by considering two distinct levels of trust. We follow this setting and consider our trust model with two trust levels: $t(u, r) = 1$ means $u$ trusts $r$ while $t(u, r) = 0$ represents $u$ distrusts $r$.

3.3. Threat scenario

To perform an inference attack, attackers are required to observe at least one onion router in a user’s onion circuit and have the knowledge of a priori trust distributions over the network. For this reason, we consider the attackers have the following attacking capabilities:

The capability of observing routers in onion circuits: In this paper, we consider attackers have two means to observe onion routers in users’ onion circuits. First, we consider the destinations (e.g., web servers) that users visit through onion circuits are certainly controlled by attackers. Attackers can employ these destinations to observe the last router in users’ circuits. Second, we consider attackers can deploy or compromise onion routers in the network. Although trust-based routing algorithms can be used to evade attackers’ routers with a high confidence, trust-based circuits are not necessarily free of attackers’ routers. Attackers can exploit their routers which are used in users’ onion circuits to observe adjacent routers in the circuits. Note that, since onion routing is not designed to defend against the attackers who can monitor the whole communication infrastructure (e.g., an ISP level attacker) (Syverson et al., 1997; Dingledine et al., 2004), attackers are assumed to have no chance to observe routers if they cannot control destinations or onion routers in users’ onion circuits.

The capability of correlating observed onion routers: Attackers can actively embed traffic watermarks or passively analyze traffic pattern using the routers and destinations under their control, and hence correlate observed onion routers in the same onion circuit (Evans et al., 2009; Murdoch and Danezis, 2005; Øverlier and Syverson, 2006; Bauer et al., 2007; Fu and Ling, 2009; Ling et al., 2009; Zhu et al., 2009). If the inference attack is performed by merely observing a single router (e.g., the last router) of onion circuits, this capability is not prerequisite.

The capability of locating the observed routers: Attackers can identify which positions the observed routers are located in onion circuits (Johnson et al., 2011). For example, since the real-world onion routing system Tor is hard coded to construct three-router onion circuits (Dingledine et al., 2004), attackers can locate the positions of the routers under their control easily. In particular, as the last router can be observed by the destination, attackers can confirm that a router stays at the last hop if this router can be observed by the destination. If attackers can observe the last router through another router under their control, this “another” router is certainly located at the second hop. Otherwise it locates at the first hop. Moreover, if onion circuits contain more than three routers, attackers can also estimate the positions of their routers by analyzing the control traffic when establishing onion circuits (Fu and Ling, 2009; Ling et al., 2009). Since attackers use the routers under their control to observe adjacent routers, if the hops of attackers’ routers can be identified, the hops of observed routers can be located as well.

The capability of accurately estimating a priori trust distributions: As discussed in Johnson et al. (2011), highly capable attackers usually have the capability of collecting the knowledge of a priori trust distributions over the network. They can make accurate estimation to reveal trust relationships among users and onion routers using outside knowledge. For example, if a user is a member of an organization, this user is more likely to trust the routers deployed by this organization. If both users and routers’ owners are members of social networks, attackers can profile the trust relationships by crawling online social networks (Gross and Acquisti, 2005; Narayanan and Shmatikov, 2009). Moreover, since trust-based onion routing algorithms are always set up by default in softwares and shared in the public, attackers who have the knowledge of a priori trust relationships can also accurately estimate users’ trust-based router selection probabilities (Johnson and Syverson, 2009; Johnson et al., 2011).
3.4 Inference attack model

We present a model for exclusively reasoning about inference attacks in the context of trust-based onion routing. Unlike generic models that consider overall anonymity, our model targets on isolating the anonymity compromised by inference attacks and therefore results in an exclusive design space for the analysis of inference attacks.

3.4.1 Model requirements
To derive the exclusive design space, our model should satisfy the following requirements:

1. Making inferences to guess the initiate user of onion circuits should be the only method that can be used to compromise anonymity in our model.
2. The design space inside of our model should be sufficient for interpreting inference attacks.
3. The design space outside of our model should be preserved.

The first requirement is used to exclude the anonymity compromised by other attacks. For example, if both the first and last routers of an onion circuit are controlled by attackers, the initiate user can be de-anonymized immediately. This case should not be considered in our model.

The second requirement expects to protect the design space of inference attacks from being affected by other attacks, even if these attacks are pre-requisites of inference attacks. For example, although attackers can benefit inference attacks by means of controlling more routers and hence observing more routers in users’ onion circuits, how attackers compromise onion routers and correlate these routers of the same circuit are out of scope of our model. As a result, given an onion circuit, which hops are observed and routers of the same circuit are out of scope of our model.

The last requirement is for eliminating the impacts on the design space outside of our model. With this requirement, our model should prevent the analysis of inference attacks from inducing side effects to the capability against other attacks, hence preserving the design space outside of our model.

3.4.2 Model design
We propose a probabilistic model to meet the three aforementioned requirements, and hence isolate the design space of inference attacks in trust-based onion routing. Our model is built on top of the trust graph presented in Section 3.2.

Definition 1. [Onion Routing Overview] We model a trust-based onion routing network as a probabilistic hypergraph $H = (U \cup R, U \times R^x)$\(^1\). $x$ is the length of onion circuits that users can make in the network. Each edge $u, C \in U \times R^x$ represents an onion circuit $C$ that is initiated by user $u$ through trust-based onion routing. $C \in R^x$ consists of $x$ routers and $C_k, 1 \leq k \leq x$ is the $k$-th router in $C$. A weight $Pr[C|u]$ is associated with the edge $(u, C)$ to represent the probability that user $u$ has to initiate the onion circuit $C$.

\(^1\) The operator $\times$ is a Cartesian product operator and $R^x$ stands for the Cartesian product of $x$ Rs.

As users initiate trust-based onion circuits by selecting routers for each hop independently, $Pr[C|u]$ can be calculated as $Pr[C|u] = \prod_{k=1}^{x} Pr[C_k|u]$. Where $Pr[C_k|u]$ represents the probability that user $u$ uses to select the router $C_k \in R$ in the $k$-th hop of his onion circuit. Trust-based onion routing algorithms determine $Pr[C_k|u]$ according to the trust level $t(u, C_k)$ and the position $k$ in the onion circuit. It can be seen, the hypergraph $H$ models trust-based onion routing by capturing a priori trust-based onion circuits distributions over the whole network.

Although the hypergraph $H$ can show the picture of the whole trust-based onion routing, it cannot well interpret the network exposed to inference attackers. Generally, inference attackers usually have the chance to only observe a partial onion circuits. As a result, we introduce a sub hypergraph $H(O) \subseteq H$ to capture the network from the view of inference attacks.

Definition 2. [Inference Attack Overview] We cut a sub hypergraph $H(O) = (U \cup R, U \times R^{(0)})$ to represent the part of trust-based onion routing that is exposed to inference attacks, where $O \subseteq \{k: 1 \leq k \leq x\}$ is the set of circuit positions observed by inference attackers and $|O| \leq \ell$ is the size of set $O$. Each edge $(u, C_0) \in U \times R^{(0)}$ represents a sub sequence of routers in $u$’s onion circuit $C$ (i.e., $C_0 = \{C_k \in O \}_{k \in C}$). These routers can be observed by inference attackers. The weight $Pr[C_0|u]$ associated with each edge $(u, C_0)$ indicates the probability that user $u$ has to construct an onion circuit with the sub sequence of routers $C_0$.

In trust-based onion routing, $Pr[C_0|u]$ can be calculated as $Pr[C_0|u] = \prod_{k \in O} Pr[C_k|u]$, because user $u$ can select the router $C_k$ in the $k$-th hop of his onion circuit according to trust $t(u, C_k)$ independently.

The set $O$ is resulted from how attackers control onion routers in users’ circuits, but makes impacts on the effectiveness of inference attacks. It could affect the independence of inference attack analysis. The hypergraph $H(O)$ apparently excludes this side effect because it considers the set $O$ as a deterministic constant input. As a result, $H(O)$ is required by our model to support the second model requirement listed in Section 3.4.1.

However, $H(O)$ is not sufficient to preserve the capability against other attacks. If we optimize $Pr[C_0|u]$ based on the hypergraph $H(O)$, it is possible to transfer selection probabilities from the routers with higher trust level to the routers with lower trust level. This side effect could induce the attacks that are prevented by trust (e.g., correlation-like attacks). To overcome this challenge, we further narrow down the hypergraph $H(O)$ by considering only the onion circuits in which the routers for each hop are equally trusted by a user. As discussed in Section 3.2, the routers that a user equally trusts have the same possibility of being controlled by this user’s attackers (Johnson and Syverson, 2009; Johnson et al., 2011).

Without loss of generality, we consider a user $u_i \in U$ who visits the Internet using trust-based onion circuits. Let $R_e \subseteq R$ be a set of routers that $u_i$ equally trusts, such as $\forall r \in R, t(u_i, r) = t_e$. Hence, $R$ can be divided into several disjoint $R_s$ with different $t_s$. That is, $R = \{R_1, R_2, \ldots, R_s\} = \{R_s \in \mathbb{R} | 1 \leq s \leq |R|\}$ and $\forall R_s, R_e \subseteq C.R(R_s \cap R_e = \emptyset)$. Where $s$ is the number of distinct trust levels and we can simply consider $t_1 > t_2 > \cdots > t_s$. Although we
set \( r = 2 \) for our experiments according to the discussion in Section 3.2, we use \( r \) to describe our model in general.

To restrict our model to the routers that \( u_i \) equally trusts and hence support the third model requirement in Section 3.4.1, we introduce the notion of hypergraph \( H(O,R^0) \) as follows.

**Definition 3. [Exclusive Design Space]** We decompose the hypergraph \( H(O) \) into several sub hypergraphs \( H(O,R^0) = (U \cup R \cup U \times R^0) \), where \( R^0 \subseteq (R_1 \cup R_2 \cup \ldots \cup R_k)^0 \). In this hypergraph, given a position \( k \in O \), all the routers in this position should be from the set \( R_k \) (i.e., \( \exists k \in O \), \( \forall C_k \in H(O,R^0) \), \( t(u_c,C_k) = t_c \)).

By using \( O \) and \( R^0 \), the hypergraph \( H(O,R^0) \) can isolate an exclusive design space for the independent analysis of inference attacks. Inference attack countermeasures that are designed based on \( H(O,R^0) \) does not necessarily sacrifice the capability against attackers’ routers. We note that \( H(O,R^0) \) is from the view of a particular user \( u_i \), and different \( u_i \) could result in different \( H(O,R^0) \) for analysis.

Now, we turn to the interpretation of inference attacks in the hypergraph \( H(O,R^0) \). According to the first model requirement from Section 3.4.1, making inferences to guess the initiate user of onion circuits is the only way to deanonymize users in our model. We then give a function to describe this kind of deanonymization in our model.

**Definition 4. [Isolated Anonymity]** Let \( Y[u_i|C_O] \) be the success probability that inference attackers have to guess the initiate user \( u_i \) by observing a sub sequence of routers \( C_O \) in \( u_i \)'s onion circuit. \( Y[u_i|C_O] \) can be calculated as:

\[
Y[u_i|C_O] = \sum_{u_U} \frac{Pr[C_O|u]}{Pr[C_O|u]} \quad (1)
\]

With Definition 4, we have two implicit assumptions: First, inference attackers have the knowledge of the isolated hypergraph \( H(O,R^0) \). This hypergraph contains sufficient a priori trust information for the calculation of \( Y[u_i|C_O] \). Second, we calculate \( Y[u_i|C_O] \) without taking unobserved hops (i.e., \( k \not\in O \)) into consideration, although these unobserved hops can benefit inference attackers with some additional information (i.e., inference attackers have the knowledge that the routers used in these unobserved hops are not controlled by them). We exclude the guessing based on these unobserved hops, because the distribution of attackers’ routers in the network can affect this guessing and hence prevent the independent analysis of inference attacks in our model. That is, the second model requirement listed in Section 3.4.1 cannot be satisfied if these unobserved hops are taken into account.

To sum up, our isolated attack model can fulfill all the three model requirements that we have presented in Section 3.4.1 using the four model definitions. In particular, the first requirement is addressed by the use of Definition 4, because this definition considers inference attacks as the only method that can be used to compromise anonymity. The second requirement is fulfilled using the Definition 2 and Definition 4. The Definition 2 isolates the portion of onion routing network that is exposed to inference attackers and thus prevents the influences from other attacking techniques. The Definition 4 expresses how to interpret inference attacks. Both of these two definitions cooperate to isolate a sufficient design space for interpreting inference attacks. The third model requirement is addressed by the Definition 3. This definition can isolate an exclusive design space for the independent analysis of inference attacks, hence preventing the impacts to the design space of other attacks.

### 4. Trust degree to the rescue

In this section, we study why trust degree is effective and how it can be used to resist inference attacks. We first analyze our inference attack model and discover trust degree as the key feature against inference attacks in Section 4.1. We then design a new routing algorithm by taking into account of trust degree to resist inference attacks in Section 4.2. We also discuss several limitations of our proposed routing algorithm in this section.

#### 4.1. Model analysis

In our inference attack model, a user \( u_i \)'s anonymity can be measured in terms of the probability \( Y[u_i|C_O] \). A lower \( Y[u_i|C_O] \) indicates inference attackers have more difficulties to deanonymize \( u_i \) by making inferences based on the observation of \( C_O \). To make the \( Y[u_i|C_O] \) prone to analysis, we separate variables from constants in Eqn. (1) as below.

\[
Y[u_i|C_O] = \frac{Pr[C_O|u]}{Pr[C_O|u]} + \sum_{u_U} \frac{Pr[C_O|u]}{Pr[C_O|u]} \quad (2)
\]

It can be seen from Eqn. (2), the calculation of \( Y[u_i|C_O] \) is apparently determined by \( \sum_{u_U} Pr[C_O|u] \) which is the only constant regardless of the probability that \( u_i \) uses to select \( C_O \) (i.e., \( Pr[C_O|u] \)). A larger \( \sum_{u_U} Pr[C_O|u] \) leads to a lower \( Y[u_i|C_O] \). As a result, given two sequences of onion routers \( C_O \) and \( C_O^\prime \), \( u_i \)'s anonymity can be better protected by using \( C_O^\prime \) rather than \( C_O \) in the context of inference attacks if \( \sum_{u_U} Pr[C_O|u] \) is larger than \( \sum_{u_U} Pr[C_O|u] \).

Given a user, we define the trust degree of a set of routers as the sum of trust or trust-based selection probabilities from other users to this set of routers. Therefore, the \( \sum_{u_U} Pr[C_O|u] \) in Eqn. (2) is the (trust) degree of the sequence of onion routers \( C_O \) in the hypergraph \( H(O,R^0) \). These probabilities are determined by trust in trust-based onion routing. According to the analysis of Eqn. (2), we find that the trust degree \( \sum_{u_U} Pr[C_O|u] \) is very effective in defeating inference attacks. We measure the effectiveness in terms of \( Y[u_i|C_O] \), the success probability that inference attackers have to guess the initiate user. A smaller \( Y[u_i|C_O] \) indicates a better anonymity protection. It can

\[\text{From } u_i \text{'s point of view, the variables are the parameters under } u_i \text{'s control, while the constants are the parameters out of } u_i \text{'s control.}\]
be seen in Eqn. (2), users who select a sequence of onion routers with a higher trust degree can have a better protection of their anonymity in the face of inference attacks, because a larger $\sum_{u \in U} \Pr[C_0|u]$ apparently results in a smaller $Y[u|C_0]$

Fig. 1 presents an example to demonstrate the effectiveness of trust degree in resisting inference attacks. In this example, Bob and Ken are two volunteers who deploy onion routers. Alice, as a user of the trust-based onion routing, trusts Bob and Ken equally. Pete is an attacker who knows a priori trust relationships among users and onion routers. If Pete observes Bob’s router in Alice’s onion circuit, he can deanonymize Alice immediately, because Alice is the only one using Bob’s router (i.e., Bob is trusted only by Alice). However, Pete cannot deanonymize Alice by observing Ken’s router in Alice’s onion circuit, he can deanonymize Alice immediately, because Alice is the only one using Bob’s router (i.e., Bob is trusted only by Alice). However, Pete cannot deanonymize Alice by observing Ken’s router immediately, because many other users can also select Ken’s router (i.e., Ken is also trusted by many other users).

In Fig. 1, Alice can be interpreted as $u_i$ in the Eqn. (2), and other users are $u \in U \setminus u_i$. Since there is only one position being exposed to inference attacks, we have $|O| = 1$. The routers deployed by Ken and Bob consist of $R^B_i$. Alice (i.e., $u_i$) should determine his probability for selecting Ken’s router or Bob’s router (i.e., $\Pr[C_0|u_i]$, $C_0 \in R^B_i$).

Although previous studies have not discovered trust degree as the key feature against inference attacks, they have proposed routing algorithms to resist inference attacks by implicitly and indirectly exploiting trust degree. For example, the downhill algorithm (Johnson et al., 2011) can enlarge trust degree by decreasing trust threshold, while social network based onion routing (Danezis et al., 2010; Mittal et al., 2013) increase trust degree through random walks on top of social networks. However, since these algorithms are not designed based on an isolated design space, they usually reduce the capability against attackers’ routers. In this paper, we conduct a very different research. Our contributions primarily lie in the disclosure of why trust degree is effective and how it can be used to defeat inference attacks without sacrificing the capability against attackers’ routers.

### 4.2 Routing algorithm with trust degree

To prove the effectiveness of trust degree in resisting inference attacks, we design a new routing algorithm by considering trust degree in this section. We investigate a user $u_i$ in the context of a population of other users whose router selection probabilities are known in advance (i.e., $\forall u \in U \setminus u_i, \Pr[C_0|u]$ are constants). Our algorithm targets on minimizing the expectation of $Y[u|C_0]$ by optimizing the distribution of $\Pr[C_0|u]$ in the hypergraph $H(O, R^B_i)$. Although there are different functions can be used to measure $Y[u|C_0]$ when optimizing the distribution of $\Pr[C_0|u]$, we choose the expectation of $Y[u|C_0]$ as our measure because previous studies used it to show the effectiveness of their trust-based onion routing algorithms (Johnson and Syverson, 2009; Johnson et al., 2013).

Let $E(Y)$ be the expectation of $Y[u|C_0]$. It can be calculated as:

$$E(Y) = \sum_{C_0 \in R^B_i} \Pr[C_0|u] \cdot Y[u|C_0].$$

(3)

We call $E(Y)$ an “expectation” although it could not satisfy $\sum_{C_0 \in R^B_i} \Pr[C_0|u] = 1$. In the hypergraph $H(O, R^B_i)$, we usually have $\sum_{C_0 \in R^B_i} \Pr[C_0|u] \leq 1$.

By targeting to minimize $E(Y)$, the design of the routing algorithm that takes into account of trust degree can be formalized as an optimization problem as follows.

$$\min E(Y), \text{s.t. } \forall k \in O, \sum_{C_n \in R^B_k} \Pr[C_n|u] = \theta_{u_k}.$$  

(4)

Where, $\theta_{u_k} \leq 1$ is the sum of probabilities that $u_i$ uses to select routers from $R_k$ for the k-th position of his onion circuit. Apparently, for $\forall k \in O$, $\sum_{i=1}^r \theta_{u_k} = 1$. $r$ is the number of distinct trust levels (see Section 3.4.2). In the isolated design space $H(O, R^B_i)$, $\theta_{u_k}$ is determined by trust-based routing algorithms and should be kept as an unchanged value when solving the optimization problem described in Eqn. (4). Since routers belonging to $R_k$ are equally trusted by $u_i$, keeping $\theta_{u_k}$ unchanged can preserve $u_i$’s capability against the routers controlled by attackers. As a result, by solving the optimization problem described in Eqn. (4) subject to $\theta_{u_k}$, we can design routing algorithm for $u_i$ to resist inference attacks without sacrificing the capability against attackers’ routers.

#### 4.2.1 Optimization for single hop router selection

To solve the optimization problem listed in Eqn. (4), we first consider the simplest scenario in which only one position of $u_i$’s onion circuit can be observed by inference attackers (i.e., $|O| = 1$). It could be the case that attackers control the destination (e.g., a web server) and observe the last router (i.e., the router in the last position $\ell$ in Fig. 2).

In this setting, the optimization problem can be simplified to the following equation:

$$\min E(Y[u|C_{\ell}]), \text{s.t. } \sum_{C_n \in R^B_{\ell}} \Pr[C_n|u] = \theta_{u_{\ell}}.$$  

(5)

Theorem 1 gives the solution of the simplified optimization problem described in Eqn. (5). The proof of this theorem can be found at Appendix A.

Fig. 1 – An example to show the effectiveness of trust degree in defeating inference attacks.

Fig. 2 – Attackers only observe the last hop in $u_i$’s circuit.
Theorem 1. Subject to $\theta_{iu}$, the expectation $E[Y(u_i|C_i)]$ that inference attackers have to deanonymize the user $u_i$ by observing the last hop (i.e., the position $t_i$) of the circuit can be minimized to:

$$\min E(Y(u_i|C_i)) = \theta_{iu} + \frac{\theta_i}{\sum_{C_i \in L_i} \sum_{u \in U_{k_i}} \Pr[C_i|u]}. $$

The corresponding optimal distribution of $\Pr[C_i|u]$ is in proportion to the trust degree of $C_i$. Given $\sum_{C_i \in L_i} \Pr[C_i|u] = \theta_{iu}$, for $\forall C_i \in L_i$, $\Pr[C_i|u] \propto \sum_{u \in U_{k_i}} \Pr[C_i|u]$. 

Theorem 1 describes the optimal router selection probability distribution over the set $R_i$ for the last position of onion circuit (i.e., the position $t_i$). By applying Theorem 1 to all the $\{R_1, R_2, ..., R_n\}$ respectively, $u_i$ can have an optimal router selection distribution over the whole router set $R$.

Steps for the use of the optimal single hop router selection algorithm: By using Theorem 1, users can apply the optimal single hop router selection strategy to existing trust-based routing algorithms with four steps:

Step 1: The user $u_i$ can first divide the set of routers $R$ into several mutually disjoint sub sets $R_s$ according to different trust level. Each $R_s$ contains routers with the same trust level $t_s$ from $u_i$’s point of view (i.e., for each $r \in R_s$, $t(u_i, r) = t_s$).

Step 2: The user $u_i$ determines $\theta_{iu}$ for each $R_s$ according to trust-based algorithms.

Step 3: For each $R_s$, $u_i$ finds the optimal distribution of $\Pr[C_i|u]$ using Theorem 1.

Step 4: After having the optimal distribution of $\Pr[C_i|u]$ for all the $R_s \subseteq R$, $u_i$ can obtain the optimal distribution of $\Pr[C_i|u]$ for $R$ by concatenating the optimal distribution of $\Pr[C_i|u]$ for all the $R_s \subseteq R$.

Using these steps, $u_i$ can minimize the expectation of being deanonymized by the inference attackers who can observe the last hop of $u_i$’s onion circuit (i.e., the position $t_i$ in the circuit), but does not sacrifice the capability of evading the routers controlled by attackers (because of keeping $\theta_{iu}$ as the same as it in trust-based algorithms). To implement this optimal router selection, the user $u_i$ is required to have the knowledge of a priori distributions that other users have to select routers for this position in their circuits.

4.2.2. Optimization for multiple hops router selection
Now, we seek the solution of the optimization problem described by Eqn. (4) in general. In this scenario, multiple positions of $u_i$’s onion circuit can be observed by inference attackers (i.e., $|O| \geq 1$). This solution can be used to derive an optimal multi-hop routing algorithm that takes trust degree into account.

Intuitively, we expect the optimal solution for multiple hops can be achieved by applying the optimal single hop solution to each hop independently. That is, we expect the optimal $\Pr[C_i|u]$ solved by Theorem 1 can be used for each hop $k \in O$ independently and hence lead to the minimized $E[Y(u_i|C_i)]$. However, this intuitive solution does not work, because the router selections for different hops are correlated.

Fig. 3 gives an example to show the correlated router selections in multiple hops. In this example, $u_i$ equally trusts routers $r_1$, $r_2$ and $r_3$. If attackers can only observe the position 3 in $u_i$’s onion circuit, we should use a larger probability to select $r_3$ than $r_2$ in the position 3, because $r_3$ is trusted by two other users (i.e., $u_1$ and $u_2$) but $r_2$ is trusted by only one (i.e., $u_3$). However, if attackers can also observe the position 2 and $u_i$ has already selected the router $r_3$ in this position (i.e., $C_3 = r_3$), the attackers can deanonymize $u_i$ immediately if $u_i$ selects $r_1$ in the position 3. The reason is that, except $u_i$, no other users trust both $r_1$ and $r_2$ in Fig. 3. As a result, to minimize inference attacks, the router selection algorithm for the position 3 should be different if the router readily used in the position 2 is different.

With no loss of generality, we consider the elements in set $O$ are indexed in an ascending order. Let $n$ be the $n$-th element in the set $O$. Let $(n−1)$ be the first $n − 1$ elements in the set $O$ and $(n−1)$ be the last $|O|−n$ elements in set $O$. As a result, $C_{(0)} = C_i$ if the $n$-th element in set $O$ is the position $k$. Moreover, $C_{(n−1)}$ denotes the sub sequence $\{C_{(1)}, C_{(2)}, ..., C_{(n−1)}\} \subseteq C_O$, while $C_{(n−1)}$ stands for the sub sequence $\{C_{(n−1)}, C_{(n−2)}, ..., C_{(0)}\} \subseteq C_O$.

For example, if $O=[1,3]$ (this indicates $|O|=2$), we have the following assignments. First, $(1)=1$ and $(2)=3$. Second, $(1)=\emptyset$, $(2)=\{1\}$ and $(3)=\{1,3\}$. Third, $(0)=\{1,3\}$, $(1)=\{3\}$ and $(2)=\emptyset$. Fourth, $C_{(0)} = C_1$ and $C_{(1)} = C_3$. Fifth, $C_{(1)} = C_2$, $C_{(2)} = C_{(1)} = C_1$ and $C_{(3)} = C_{(1)}, C_{(2)} = C_1, C_3$. And last, $C_{(0)} = C_{(1)}, C_{(2)} = C_{(3)}, C_{(1)} = C_3$ and $C_{(2)} = C_3$.

As the router selections for multiple hops are correlated, we cannot calculate $\Pr[C_i|u]$ using $\Pr[C_i|u] = \prod_{O \in U} \Pr[C_i|u]$, because this calculation assumes the multi-hop router selections are independent. Instead, $\Pr[C_i|u]$ can be calculated in a more general version as follows:

$$\Pr[C_i|u] = \prod_{n=1}^{O} \Pr[C_{(n)}|u, C_{(n+1)}]. $$

Where $\Pr[C_{(n)}|u, C_{(n+1)}]$ represents the conditional probability that the user $u$ uses to select the router $C_{(n)}$ for the $(n)$-th position in his onion circuit in case the user $u$ has already selected routers $C_{(n+1)}, C_{(n+2)}, ..., C_{(O)}$ for the $(n+1), (n+2), \cdots, (|O|)$-th positions in the circuit respectively.

Based the Eqn. (6), the optimization problem that is described in Eqn. (4) can be transformed to a general version as:

$$\min E(Y), \text{s.t.} \ \forall (n) \in O, \ \sum_{C_{(n)} \in R_{(n)}} \Pr[C_{(n)}|u, C_{(n+1)}] = \theta_{(n)k}. $$

![Fig. 3 – Router selections in multiple hops are correlated.](image)
Note that, since trust-based onion routing algorithms implement router selections for multiple hops according to trust independently, \( Pr[C_{0}[u|u, C_{n-3}] = Pr[C_{0}[u|u] \text{ if the user } u \text{ uses trust-based routing algorithms. To derive inference attack countermeasures without tradeoffs, we find the solution of Eqn. (7) in the isolated design space } H(O, R^{(O)}) \text{, in which the } \theta_{0k} \text{ is pre-defined when } u \text{ runs trust-based routing algorithms (i.e., } \theta_{0k} = \sum_{C_{0} \in uR_{e}} Pr[C_{0}|u|u] \text{). This value of } \theta_{0k} \text{ is kept as a constant in the design space } H(O, R^{(O)}) \text{. As a result, in the process of solving the optimization problem described in Eqn. (7), for a given } n \text{, no matter which routers } C_{0-1}, C_{n-2}, \ldots, C_{00} \text{ have been readily used in the positions } (n+1), (n+2), \ldots, (O), \text{ the summarized probability that } u \text{ uses to select routers } C_{0} \in R_{e} \text{ cannot be changed. That is, } \sum_{C_{0} \in uR_{e}} Pr[C_{0}|u|C_{n-1}] \text{ should be continuously equal to the constant } \theta_{0k} \text{ that is pre-determined by trust-based routing algorithms. }

Theorem 2 gives the solution of the general optimization problem described in Eqn. (7). The proof of this theorem is presented at Appendix B.

**Theorem 2.** Subject to \( \theta_{0k} \Omega 1 \leq n \leq |O| \), the expectation \( E[Y[u|C_{0}]] \text{ that inference attackers have to deanonymize the user } u \text{ by observing a sub sequence } C_{0} \text{ of the onion circuit can be minimized to:}

\[
\min E[Y] = \frac{\prod_{j=1}^{O} \theta_{0j}}{\prod_{n=1}^{O} \theta_{0k} + \sum_{C_{0} \in uR_{e}} \sum_{u \in U[u]} Pr[C_{0}|u]}
\]

Since the selection order makes no effects to the solution of the optimization problem, we simply give the optimal distribution for \( \forall \Omega 1, Pr[C_{0}[u|u, C_{n-1}]] \) by considering \( u \) selects routers \( C_{0} \) in an descending order (i.e., selecting \( C_{0} \) in the order from \( n = |O| \) to \( n = 1 \)). In this case, when we consider \( Pr[C_{0}[u|u, C_{n-1}]] \), the routers \( C_{0-1}, C_{n-2}, \ldots, C_{0} \) that are used in the positions \( (n+1), (n+2), \ldots, (O) \) are readily known (i.e., \( C_{n-1} \) is a known value). The optimal distribution of \( Pr[C_{0}|u|C_{n-1}] \) for each \( (n) \in \{ 0 \} \) is as below.

\[
Pr[C_{0}|u|u, C_{n-1}] = \theta_{0|n|}, \forall \Omega 0 \in R_{e}:
\]

**Theorem 2** describes the optimal probability distributions for selecting the sequence of routers \( C_{0} \) from \( R^{(O)} \). By implementing **Theorem 2** to all the \( \{ R_{1}, R_{2}, \ldots, R_{|O|} \} \) respectively, \( u \) can have an optimal distribution for selecting the sequence of routers \( C_{0} \) over the whole set \( R^{(O)} \).

**Theorem 2** derives an optimal routing algorithm. This algorithm does not make any tradeoffs between the capability against inference attacks and the capability against attackers’ routers. It can be embedded into existing trust-based routing algorithms to further enhance the anonymity protection.

**Steps for the use of the optimal multi hops router selection algorithm:** By using **Theorem 2**, users can apply the optimal multi hops router selection strategy to existing trust-based routing algorithms with four steps:

1. **Step 1:** The user \( u_{i} \) can first divide the set of routers \( R \) into several mutually disjoint sub sets \( R_{s} \) according to different trust level \( t_{s} \)s, such as \( R = \{ R_{1}, R_{2}, \ldots, R_{s} \} \). Given the set of circuit positions \( O \) observed by inference attackers, \( u_{i} \) can choose \( R^{(O)} = \{ R_{1}, R_{2}, \ldots, R_{s} \} \), where each \( R_{s} \in R^{(O)} \) contains routers with the same trust level \( t_{s} \) from \( u_{i} \)'s point of view (i.e., for each \( r \in R_{s}, t(u_{i}, r) = t_{s} \)).

2. **Step 2:** The user \( u_{i} \) determines \( \theta_{0k} \) for any \( (n) \) and \( R_{s} \in R^{(O)} \) according to trust-based routing algorithms.

3. **Step 3:** For each \( R^{(O)} \), \( u_{i} \) finds the optimal distribution of \( Pr[C_{0}|u] \) using **Theorem 2**.

4. **Step 4:** After having the optimal distribution of \( Pr[C_{0}|u] \) for all the \( R^{(O)} = \{ R_{1}, R_{2}, \ldots, R_{s} \} \), \( u_{i} \) can obtain the optimal distribution of \( Pr[C_{0}|u] \) for \( \{ R_{1}, R_{2}, \ldots, R_{s} \} \) by concatenating the optimal distribution of \( Pr[C_{0}|u] \) for all the \( R^{(O)} = \{ R_{1}, R_{2}, \ldots, R_{s} \} \).

Using these steps, \( u_{i} \) can minimize the expectation of being deanonymized by the inference attackers who can observe the set of positions \( O \) in onion circuit. To use our algorithm, the user \( u_{i} \) is required to have the knowledge of which positions can be observed by attackers (i.e., the set \( O \)), as well as a priori distributions that other users have to select routers for these positions in their onion circuits (i.e., \( Pr[C_{0}|u, C_{n-1}] \) for \( (n) \in O \) and \( \forall \Omega u_{i} \)).

4.2.3. Algorithm limitation

We use the optimal solution described in **Theorem 1** and **Theorem 2** as a proof of concept routing algorithm to show the effectiveness of trust degree in theory, although we acknowledge this algorithm contains several limitations in practice. In this section, we describe these limitations and discuss possible solutions to work around them. This discussion gives a perspective of practical algorithms in the future.

**Large requirements of a priori knowledge:** The user \( u_{i} \) who selects routers using our algorithm is required to have the knowledge of the hypergraph \( H(O, R^{(O)}) \). This knowledge is two-fold. One is the set \( O \) and the other is \( Pr[C_{0}|u|C_{n-1}] \) for \( (n) \in O, \forall \Omega u_{i} \). We note that inference attackers are also required to have the knowledge of \( Pr[C_{0}|u, C_{n-1}] \). As these inference attackers are deemed to exist, the users who have the same knowledge are certainly existing as well. In addition to \( Pr[C_{0}|u, C_{n-1}] \), these users also need to know which positions of onion circuits can be observed by attackers (i.e., the set \( O \)). This requirement is practical and depends on how users construct onion circuits according to trust. For example, if users select routers with high level trust for all the positions in their onion circuits, the set \( O \) is likely to only contain the last position (i.e., \( O = \{ t \} \)). However, if users adopt the downhill algorithm for router selection and the trust thresholds after the \( k \)-th position is low enough, \( O \) could contain the last \( k + 1 \) positions with a high probability (i.e., \( O = \{ k, k + 1, \ldots, t \} \)). Due to the large requirements of a priori knowledge about the onion routing network, the users who can benefit from our optimal solution could be very few (i.e., the number of users who can apply **Theorem 1** and **Theorem 2** is very small). They are usually the government users or security officers. Compared with normal users, these favored users could be highly capable of collecting the information about other users and require stronger anonymity protection.
Deadlock problem: We design our algorithm by investigating a user \( u_i \) in the context of a population of other users whose router selection probabilities are already known. This consideration implicitly assumes that other users’ router selection probabilities are stable. However, this assumption does not hold if there are more than one user who adopts our algorithm in the network. For example, consider two users \( u_i \) and \( u_j \) who could select the router \( C_k \) for the \( k \)-th position of their circuits using our algorithm. The probability \( \Pr[C_k|u_i] \) that \( u_i \) uses to select \( C_k \) is determined according to the probability \( \Pr[C_k|u_i'] \) that \( u_i' \) uses to select \( C_k \), while the \( \Pr[C_k|u_i'] \) on the other hand is calculated based on \( \Pr[C_k|u_i] \). Therefore, a deadlock occurs. To work around this issue, we suggest two possible solutions. First, each user \( u_i \) who adopts our algorithm can initiate \( \Pr[C_k|u_i] \) by assuming \( \Pr[C_k|u_i'] \) for \( u_i' \in U \setminus u_i \), are calculated using trust-based algorithms at first, and then update \( \Pr[C_k|u_i] \) periodically afterwards. When \( u_i \) updates \( \Pr[C_k|u_i] \), he is required to know the \( \Pr[C_k|u_i] \) of the users \( u_i' \) who also adopt our algorithm in current time period. This can be achieved by sharing the \( \Pr[C_k|u_i] \) among all the users \( u_i' \) who adopt our algorithm in each period. By this way, all the users who adopt our algorithm can work together to play a race condition game, and their router selection probabilities approximate to the optimal values as time passing. Second, we can adjust our algorithm to let user \( u_i \) select routers according to other users’ trust rather than other users’ selection probabilities. For example, Theorem 1 can use \( \Pr[C_k|u_i] \approx \sum_{u_j \in U \setminus u_i} \Pr(C_k|u_j) \) to replace \( \Pr[C_k|u_i] \approx \sum_{u_j \in U \setminus u_i} \Pr(C_k|u_j) \). This method could cause the optimal solution degenerating to a near-optimal solution.

Impacts on other users: When the user \( u_i \) adopts our algorithm, he could reduce the probability of selecting some routers. Consider the probability \( \Pr[C_k|u_i] \) that \( u_i \) uses to select the router \( C_k \) is reduced, the other users who also trust and select \( C_k \) will become more likely to be guessed through \( C_k \). We argue this impact in two aspects. First, the government users or security officers who have the knowledge of \( \Pr[C_k|u_i] \) directly adopt our algorithm to avoid this impact. Second, since the users who can run our algorithm in the network are few, the selection probability reduction of a particular router is usually small and therefore makes a little impact. Moreover, compared with the users who can adopt our algorithm (they are usually government users or security officers), the normal users require relatively weak anonymity protection. As a result, although these normal users cannot avoid this impact using our algorithm, they could tolerate such a little impact.

Although our optimal algorithm suffers from the three aforementioned limitations, we still choose it to investigate the effectiveness of trust degree because: 1), as a proof of concept routing algorithm, it provides the upper bound of trust degree’s effectiveness; 2), these limitations can be worked around or only make tolerable impacts.

5. Investigating the effectiveness of trust degree

In this section, we investigate the effectiveness of trust degree in resisting inference attacks using two real-world social networking datasets from Viswanath et al. (2009). We show this effectiveness by comparing \( E(Y) \) between a raw routing algorithm with and without considering trust degree. The raw algorithm with considering trust degree represents the algorithm embedded with our optimal solution (see Theorem 1 and Theorem 2), while the raw algorithm without considering trust degree is the algorithm itself. In this investigation, we consider two raw algorithms. One is the traditional trust-based onion routing algorithm that is proposed to defeat correlation-like attacks by Johnson and Syverson (2009) and Johnson et al. (2011), and the other is the downhill algorithm that is proposed to mitigate inference attacks by Johnson et al. (2011).

The organization of this section is as follows. We first describe the two real-world datasets that we use for our investigation in Section 5.1. We then investigate trust degree’s effectiveness based on the traditional trust-based routing algorithm and the downhill algorithm in Sections 5.2 and 5.3, respectively.

### 5.1 Datasets

We employ two public social networking datasets (Viswanath et al., 2009) for our investigation. The authors of these two datasets crawled the New Orleans regional network in Facebook from December 29th, 2008 to January 3rd, 2009. One dataset contains a friendship graph and the other is an interaction graph. In a friendship graph, a directed edge represents a person is recorded in another person’s friend list in Facebook. While in an interaction graph, if there is a directed edge, beside that a person must be another person’s friend, these two people also need to have direct communications through social medias (e.g., post walls in Facebook). Recent studies (Viswanath et al., 2009; Wilson et al., 2009) argued that the interaction graph can better represent the trust among human beings than the friendship graph in the real world. However, to investigate the effectiveness of trust degree in diverse trust settings, we conduct our evaluation using both of the two graphs.

In these two datasets, we regard the start point of each directed edge as a user and the end point of the edge as a volunteer who deploys an onion router. We therefore map the friendship graph or the interaction graph to a trust graph \( G = (U \cup R, \exists R) \), where \( U \) is the set of users and \( R \) is the set of routers (or the set of routers’ owners). Note that, a person can be a user and a router’s owner simultaneously (i.e., \( U \cap R \neq \emptyset \)). As discussed in Section 3.2, we consider two distinct levels of trust in our evaluation (i.e., \( v = 2 \)). In particular, for \( Y(u, r) \in U \times R \), if the edge \( (u, r) \) exists in the dataset, we consider the user \( u \) trusts the router \( r \) (i.e., \( t(u, r) = 1 \)). Otherwise, we regard the user \( u \) distrusts the router \( r \) (i.e., \( t(u, r) = 0 \)).

### Table 1 – The basic statistics of the two datasets (Viswanath et al., 2009).

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of users</th>
<th># of routers</th>
<th># of edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>friendship graph</td>
<td>53,609</td>
<td>63,406</td>
<td>1,539,193</td>
</tr>
<tr>
<td>interaction graph</td>
<td>30,988</td>
<td>37,467</td>
<td>262,684</td>
</tr>
</tbody>
</table>

As discussed in Section 3.2, we consider two distinct levels of trust in our evaluation (i.e., \( v = 2 \)). In particular, for \( Y(u, r) \in U \times R \), if the edge \( (u, r) \) exists in the dataset, we consider the user \( u \) trusts the router \( r \) (i.e., \( t(u, r) = 1 \)). Otherwise, we regard the user \( u \) distrusts the router \( r \) (i.e., \( t(u, r) = 0 \)).
We exclude the users who trust only one router from our evaluation, because trust-based routing algorithms cannot benefit from trust degree for these users. Table 1 summarizes basic statistics of our evaluated datasets which do not include the users who trust only one router.

5.2. Trust-based algorithm benefits from trust degree

In this section, we investigate the effectiveness of trust degree by embedding our optimal solution into the traditional trust-based onion routing algorithm. This algorithm is proposed by Johnson and Syverson (2009) and Johnson et al. (2011) and can be used to construct trust-based onion circuits to defeat correlation-like attacks. In our investigation, we consider each user in the real-world social networking datasets (listed in Table 1) as one by one. Each point in the figures represents a user that we consider as ui in the datasets. If a point is located at the left-top side of the red diagonal line, it indicates this user’s anonymity can be better protected against inference attacks.

Given a user ui, the set of routers R can be divided into two disjoint sub sets R1 and R2 according to the two distinct trust levels t1 = 1 and t2 = 0. ∀ r ∈ R1, t(u, r) = t1 = 1 while ∀ r ∈ R2, t(u, r) = t2 = 0. Using the trust-based algorithm, the user ui can select the routers from R1 uniformly at random but cannot use the routers from R2 in any positions of his onion circuits (Johnson and Syverson, 2009; Johnson et al., 2011). For this reason, ui has the same probability 1/jR1j for selecting each router from R1.

When embedding our optimal solution (see Theorem 1 and Theorem 2) into the trust-based algorithm, we have θk1 = 1 and θk2 = 0 where k ∈ O. Note that, as our optimal solution enables ui to select routers according to other users’ router selection distributions (i.e., Pr[C0|u] for ∀ u ∈ U), we consider other users employ the trust-based algorithm for their router selections.

Fig. 4 illustrates the distributions of E[Y|u|C0] when ui uses trust-based algorithm (shown in y-axis) and this algorithm embedded with our optimal solution (shown in x-axis). Each point in the figures represents a user that we consider as ui in the datasets. If a point is located at the left-top side of the red diagonal line, it indicates this user’s anonymity can be better protected against inference attacks by taking advantage of our optimal solution. It can be seen in Fig. 4, all the points are located at the left-top side of the red diagonal lines regardless
The two cases for the three observed positions. The maximum more than 99.6% users have degree (i.e., 100% users have smaller, or at least the same, $E(Y_u|C_1)$ compared with the trust-based algorithm which does not take advantage of trust degree (i.e., 100% users have $\Delta E \geq 1$). Moreover, for more than 99.6% users, trust degree can help them reduce $E(Y_u|C_0)$ (i.e., more than 99.6% users have $\Delta E > 1$). The largest reduction is up to 340.6 times (i.e., max($\Delta E$) = 340.6). Our results confirm that trust-based algorithm can benefit a lot from trust degree in resisting inference attacks under real-world settings.

Table 2 – The maximum $\Delta E$ and the percentage of users who meet a particular condition of $\Delta E$ in each graph.

| Datasets and $|O|$ | $\Delta E$ | $\Delta E \geq 1$ | $\Delta E > 1$ | $\Delta E \geq 2$ | $\Delta E > 10$ |
|-------------------|-----------|------------------|---------------|-----------------|-------------|
| Friendship graph, $|O| = 1$ | 31.1 | 100% | 99.6% | 37.4% | 0.4% |
| Friendship graph, $|O| = 2$ | 340.6 | 100% | 100% | 85.4% | 17.1% |
| Friendship graph, $|O| = 3$ | 186.4 | 100% | 100% | 85.9% | 7.5% |
| Interaction graph, $|O| = 1$ | 15.2 | 100% | 99.8% | 35.5% | 0.8% |
| Interaction graph, $|O| = 2$ | 61.8 | 100% | 100% | 77.7% | 5.7% |
| Interaction graph, $|O| = 3$ | 33.6 | 100% | 100% | 78.1% | 0.8% |

5.3. Downhill algorithm benefits from trust degree

In this section, we investigate the effectiveness of trust degree by embedding our optimal solution into the downhill algorithm. The downhill algorithm is proposed by Johnson et al. (2013) and can be used to mitigate inference attacks by sacrificing the capability of evading attackers’ routers. The same as we have done in Section 5.2, we consider all the users in the two datasets as to be $u_l$ one by one. We compare $E(Y_u|C_0)$ among the trust-based algorithm, the downhill algorithm and the downhill algorithm embedded with our optimal solution. A smaller $E(Y_u|C_0)$ in a particular algorithm means this algorithm can better protect anonymity against inference attacks.

Unlike trust-based algorithm which limits $u_l$ to only select the routers $u_i$ trusts (i.e., $u_i$ has the probability to select $r$ if and only if $t(u_i, r) = t_1 = 1$), the downhill algorithm enables $u_l$ to select routers uniformly at random from sets with a decreasing trust threshold along onion circuits (Johnson et al., 2011). Let $t(n)$ be the trust threshold used in the position $(n) \in O$ of $u_l$’s onion circuits, where $(n)$ represents the $n$-th element in the observed position set $O$. Using the downhill algorithm, we have $t(n) \leq t(n')$ if the position $(n)$ is closer to $u_l$ than the position $(n')$ (i.e., $(n') < (n)$). Since we consider only two distinct trust levels in our evaluation (i.e., $t_1 = 1$ and $t_2 = 0$), $u_l$ uses the probability $1/(R_1)$ to select routers from $R_1$ for the position $(n) \in O$ if $t(n) = 1$, and uses the probability $1/R$ to select routers from $R$ for the position $(n) \in O$ if $t(n) = 0$.

When embedding our optimal solution (see Theorem 1 and Theorem 2) into the downhill algorithm, we have $\theta_{(n)} = 1, \theta_{(n)} = 2$ for the position with threshold $t(n) = 1$, and $\theta_{(n)} = |R_1|/|R|, \theta_{(n)} = |R_2|/|R|$ for the position with threshold $t(n) = 0$. Where $R_1$ is the set of routers with $\forall r \in R_1, t(u_i, r) = t_1 = 1$ and $R_2$ is the set of routers with $\forall r \in R_2, t(u_i, r) = t_2 = 0$.

In this evaluation, we consider three positions of onion circuits can be observed by inference attackers (i.e., $|O| = 3$). According to the number of observed positions using trust threshold $t(n) = 1$, we have two different cases in our evaluation. Table 3 lists these two cases. The case ① has only one observed position with $t(n) = 1$ while the case ② has two observed positions with $t(n) = 1$.

Fig. 5 shows the $E(Y_u|C_0)$ in the downhill algorithm embedded with our optimal solution, the downhill algorithm and trust-based algorithm for each $u_i$ in the two datasets listed in Table 1. It can be seen, although the downhill algorithm can better protect anonymity against inference attacks than trust-based algorithm, our optimal solution can be embedded into the downhill algorithm to further improve the effectiveness of resisting inference attacks. Unlike the downhill algorithm that sacrifices the capability against attackers’ routers to mitigate inference attacks, our optimal solution makes no tradeoffs because it takes advantage of trust degree among equally trusted routers.

To sum up, we have successfully demonstrated that trust degree is very effective in resisting inference attacks and can be used to further enhance the protection of anonymity in trust-based onion routing under real-world settings.

6. Discussion

In this section, we discuss two design assumptions of this paper. One is related to the design of our inference attack
model in Section 6.1, while the other is relevant to the correct and accurate collection of trust degree to defeat inference attacks in Section 6.2.

### 6.1. Equal trust and other attacks

In the design of our inference attack model, we claim that considering trust degree among the routers with equal trust can preserve the capability against other attacks. To explain the reason of this claim, we analyze three major "other" attacks and elaborate on why equal trust can preserve the capability against these attacks in trust-based onion routing.

- **Correlation-like attack** (Evans et al., 2009; Murdoch and Danezis, 2005; Øverlier and Syverson, 2006; Bauer et al., 2007; Fu and Ling, 2009; Ling et al., 2009; Zhu et al., 2009; Hopper et al., 2010): This kind of attack is the most popular attack against onion routing networks. The use of trust-based routers is for defeating correlation-like attacks, because this kind of attacks cannot be performed in the onion circuits which do not contain attackers’ routers. As a result, the capability against correlation-like attacks can be determined by the capability against attackers’ routers. Since the routers with equal trust have the same probability of being controlled by attackers, these routers have the same capability against correlation-like attacks.

- **Predecessor attack** (Wright et al., 2004): This attack is first proposed to compromise anonymity in Crowds networks (Reiter and Rubin, 1998), but later research (Wright et al., 2004) has proved its effectiveness in attacking onion routing networks as well. The basic idea of this attack is to record the predecessor of attackers’ routers in multiple onion connections and the initiate user is likely to appear in those recorded predecessors. As a result, the capability against the predecessor attack can be determined by the capability of preventing the use of attackers’ routers as the first router in onion circuits. Therefore, the routers with equal trust share the same capability against the predecessor attack.

- **Node degree attack** (Mittal et al., 2013): Since attackers can arbitrarily forge trust relationships among themselves, they can perform the node degree attack by amplifying trust degrees of the routers under their control (Mittal et al., 2013). This attack poses a significant threat to the social network based onion routing through conventional random walks (Danezis et al., 2010). However, by considering trust degree among the routers with equal trust, this attack can be...
prevented through the use of trust before the use of trust degree. As the routers with equal trust have the same probability to have forged trust degree, they preserve the same capability against the node degree attack.

6.2. The correctness and accuracy of trust degree

Although we design our trust degree based algorithm by assuming the correct and accurate information of a priori trust distributions are already known, we acknowledge that this assumption is only made available for the theoretical design. Practical implementations of our proposed algorithm require feasible solutions to collect the correct and accurate information of a priori trust distributions. However, we mainly focus on the discovery of trust degree as a key feature against inference attacks in this paper. How to correctly and accurately collect the information of a priori trust distributions is out of scope of this paper. We leave this research topic in the future work.

7. Conclusions

In this paper, we uncover trust degree as a key feature against inference attacks for the first time. The discovery of trust degree is based on the analysis of an isolated inference attack model. To demonstrate the effectiveness of trust degree in resisting inference attacks, we propose a proof of concept routing algorithm. We have evaluated the proposed algorithm using real-world social networking datasets, and demonstrated that trust degree is very effective against inference attacks. In contrast to prior studies, we have conducted a very different research by building and analyzing an isolated attack model in this paper. Our findings provide evidence to confirm the usefulness of the new research principle.

Appendix A. The proof of Theorem 1

Proof. We first consider a simple optimization problem as follows.

\[
\min \left( x \frac{x}{x + a} + y \frac{y}{y + b} \right) \quad \text{s.t. } x + y = \beta.
\]

Where \( x \) and \( y \) are variables while \( a \) and \( b \) are constants. Let \( f(x) = x - x(\beta - x)/(\beta - x + b) \). Then the above optimization problem can be simplified as:

\[
\min f(x) = \min \left( x - x \frac{x}{x + a} + (\beta - x) \frac{\beta - x}{\beta - x + b} \right).
\]

It is known that, if \( f(x) \) has an extreme value and \( f(x) \)'s second derivative is larger than 0, this extreme value is \( f(x) \)'s minimum value. And this minimum value can be obtained by letting \( f(x) \)'s first derivative equals to 0. Such that, if \( f'(x) = d^2 f(x)/dx^2 > 0 \), \( f(x) \) can reach its minimum value when \( f'(x) = df(x)/dx = 0 \).

When \( \beta = x + y > x, a > 0 \) and \( b > 0 \), we can prove that \( f''(x) > 0 \). In this case, we find the minimum value of \( f(x) \) by solving the quadratic equation \( f'(x) = 0 \).

We have two roots of \( f'(x) = 0 \). One is a positive value and the other is a negative value. We only consider the positive result, \( x = \beta \cdot a/(a + b) \). Applying this \( x \) to \( f(x) \), we have:

\[
\min f(x) = \min \left( x - x \frac{x}{x + a} + (\beta - x) \frac{\beta - x}{\beta - x + b} \right) = \frac{\beta^2}{a + b + 1}.
\]

Since \( x + y = \beta \), we have:

\[
\frac{x^2}{x + a} + \frac{y^2}{y + b} \geq \frac{\beta^2}{a + b + 1} = \frac{(x + y)^2}{a + b + 1}.
\]

To satisfy the equality, we have \( x = \beta \cdot a/(a + b) \) and \( y = \beta \cdot b/(a + b) \) (i.e., \( x \sim a \) and \( y \sim b \)).

Now, we turn back to the optimization problem described in Eqn. (5). By iteratively applying the inequality (A.1) to \( E(Y | u_i | C_i) \), we have:

\[
E(Y | u_i | C_i) \geq \min \left( \sum_{C \in \mathcal{C}_u} \Pr(C_i | u_i) \cdot Y | u_i | C_i \right)
\]

To satisfy all the equalities and achieve the minimal value, we have \( \forall C_i \in \mathcal{C}_u, \Pr(C_i | u_i) \cdot \sum_{u \in U | u_i} \Pr(C_i | u) \) subject to \( \theta_{u} = \sum_{C \in \mathcal{C}_u} \Pr(C_i | u) \).

Therefore, Theorem 1 is proved. \( \square \)

Appendix B. The proof of Theorem 2

Proof. We first consider a function as follows.

\[
\sum_{C_i \in \mathcal{C}_u} \prod_{n=1}^{\infty} \frac{\prod_{n=1}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2})^2}{\prod_{n=2}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2}) + \prod_{n=2}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2})}, \quad \text{(B.1)}
\]

subject to \( \sum_{C_i \in \mathcal{C}_u} \Pr(C_i | u_i, C_{i+1}) = \theta_{u_i} \). As the operator \( \sum_{C_i \in \mathcal{C}_u} \) is irrelevant to \( \prod_{n=2}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2}) \), we can consider it as a constant in the function (B.1). Simply let \( \Omega = \prod_{n=2}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2}) \). By applying the inequality (A.2) to the function (B.1), we can prove the minimal value of the function (B.1) as below.

\[
\theta_{u_i} \Omega + \sum_{C_i \in \mathcal{C}_u} \sum_{u \in U | u_i} \prod_{n=1}^{\infty} \Pr(C_{i_1} | u_i, C_{i_2}) \quad \text{(B.2)}
\]
To achieve this minimum value, for $\forall C(1) \in R_n$ subject to $\sum_{C_{i_1} \in R_i} Pr[C_{i_1} | u_i, C_{i_1}] = \theta_{i_1}$, we have:

$$Pr[C_{i_1} | u_i, C_{i_1}] = \sum_{C_{i_2} \in R_i} \sum_{u_i \in U(u_i)} \sum_{n=1}^{M} Pr[C_{i_2} | u_i, C_{i_2}]$$

where, $C_{i_1} = \emptyset$ and $R_i^1 = \emptyset$.

Now, we turn back to the optimization problem described in Eqn. (7).

$$E(Y[u_i | C_0]) = \sum_{n=1}^{M} \frac{Pr[C_{i_1} | u_i]^2}{Pr[C_{i_1} | u_i] + \sum_{u_i \in U(u_i)} Pr[C_{i_2} | u_i]}$$

where, $Pr[C_0 | u_i] = \prod_{n=1}^{M} Pr[C_{i_1} | u_i, C_{i_1}]$, and the operator $\sum_{C_{i_1} \in R_i^0}$ can be extended to $\sum_{C_{i_2} \in R_i^0} = \sum_{C_{i_3} \in R_i^0} \cdots \sum_{C_{i_2} \in R_i^0}$.

By considering that each operator $\sum_{C_{i_1} \in R_i}$ is irrelevant to $\prod_{n=1}^{M} Pr[C_{i_1} | u_i, C_{i_2}]$, we can iteratively apply the similar optimization process like (B.2) and (B.3) to $E(Y[u_i | C_0])$, and therefore prove:

$$\min \ E(Y) = \sum_{n=1}^{M} \frac{\theta_{i_1}^n}{\prod_{n=1}^{M} \theta_{i_1}^n}$$

To reach this minimal value, for $\forall C(0) \in R_n$ subject to $\sum_{C_{i_1} \in R_i} Pr[C_{i_1} | u_i, C_{i_1}] = \theta_{i_1}$, we have:

$$Pr[C_{i_1} | u_i, C_{i_1}] = \sum_{C_{i_2} \in R_i} \sum_{u_i \in U(u_i)} \prod_{n=1}^{M} Pr[C_{i_2} | u_i, C_{i_2}]$$

Therefore, Theorem 2 is proved. □

References


Fu Xinwen, Ling Zhen. One cell is enough to break Tor’s anonymity. In: Proc. Black Hat DC 2009.


Peng Zhou received the BSc and MSc degrees in Electrical Engineering from Donghua university in 2004 and 2007, respectively. He then joined Hewlett-Packard Company, Shanghai, China and worked as a firmware engineer and security software engineer for three years. He is now working toward the PhD degree in the Department of Computing at the Hong Kong Polytechnic University. His research interests cover network security and privacy.
Xiapu Luo received his Ph.D. degree in Computer Science from the Hong Kong Polytechnic University in 2007 and then spent two years at the Georgia Institute of Technology as a post-doctoral research fellow. He is now a research assistant professor in the Department of Computing at the Hong Kong Polytechnic University. He is also a researcher affiliated with the Shenzhen Research Institute of the Hong Kong Polytechnic University. His current research focuses on network security and privacy, Internet measurement, and Smartphone security.

Rocky K.C. Chang received the PhD degree in computer engineering from Rensselaer Polytechnic Institute. Immediately after that, he joined the IBM Thomas J. Watson Research Center working on performance analysis and simulation tools. He then joined the Department of Computing at the Hong Kong Polytechnic University, where he is now an associate professor. He is leading an Internet Infrastructure and Security Laboratory, addressing problems in network security, network measurement, and network operations and management. He is a member of the IEEE and ACM.