Scalable Anonymous Communication with Provable Security

Prateek Mittal, Nikita Borisov[*]
Dept. of Electrical and Computer Engineering,
U. of Illinois at Urbana-Champaign,
{mittal2,nikita}@illinois.edu

Carmela Troncoso, Alfredo Rial[†]
ESAT/COSIC,
IBBT-K.U.Leuven,
firstname.lastname@esat.kuleuven.be

Abstract

A key problem in Tor's architecture is that it requires users to maintain a global view of the system, which will become costly as the size of the network increases. Several peer-to-peer approaches have been proposed in order to alleviate the scalability concerns of the Tor network, but they are only able to provide heuristic security; in fact, the security community has been quite successful at breaking the state of the art systems using both passive and active attacks. In this paper, we explore new primitives for scalable anonymous communication, with a focus on providing provable security guarantees. First, we propose a new approach for secure peer-to-peer anonymous communication based on a reciprocal neighbor policy. Secondly, we propose PIR-Tor, a client-server scalable architecture for anonymous communications based on Private Information Retrieval.

1 Introduction

In this era of pervasive surveillance, our online activities are recorded, aggregated, and analyzed. Anonymous communication is a basic privacy enhancing technology that hides the identity of communicating partners from third parties, or user identity from the remote party. The Tor network [7] is a deployed system for anonymous communication that serves hundreds of thousands of users [12]. Tor is used to protect the privacy of journalists, dissidents, whistle-blowers, law-enforcement and even government embassies [10].

A key problem in Tor's architecture is that it requires users to maintain a global view of the system in order to allow them to randomly choose relays. As the size of the network increases, maintaining a global view of the system becomes costly. McLachlan et al. [13] show that in the near future, the Tor network could be spending an order of magnitude more bandwidth maintaining this global view than relaying anonymous communication.

In order to alleviate the scalability concerns of the Tor network, several peer-to-peer approaches have been proposed [13, 17, 15, 19, 20, 23]. However, all of the proposed peer-to-peer approaches only provide heuristic security; in fact, the security community has been quite successful at breaking the state of the art systems using both passive and active attacks [1, 4, 6, 16, 26]. Moreover, these systems are quite complex, and are limited in their applicability as they only work with structured topologies. In this paper, we explore new primitives for scalable anonymous communication, with a focus on providing provable security guarantees.

First, we propose a new approach for peer-to-peer anonymous communication based on *reciprocal neighbor policy*, that works even with unstructured topologies. Our key insight is to entangle the fingertables of neighboring nodes in the topology, i.e., if a malicious node X tries to exclude an honest node Y from its fingertable, then the honest node Y will also exclude the malicious node X from its fingertable. We formally prove that a reciprocal neighbor policy does not give any advantage to the adversary.We also outline mechanisms for securing the reciprocal neighbor policy itself, for both unstructured and structured topologies.

Secondly, we propose PIR-Tor, a client-server architecture for scalable anonymous communication that can tolerate any fraction of compromised relays in the network. Our main idea is to leverage private information retrieval techniques (PIR) to query central directory servers for a few random relays in the network. Querying only a few nodes is the key to the scalability of our architecture, while the use of PIR techniques safeguards clients against passive attacks by malicious servers.

[*]These authors were supported in part by National Science Foundation Grant CNS 06-27671.

[†]These authors were supported in part by the IAP Programme P6/26 BCRYPT of the Belgian State, and are funded by the Research Foundation - Flanders (FWO).

2 Background and Related Work

Tor [7] is a popular low-latency anonymous communication system, which serves hundreds of thousands of users every day [12]. Each Tor client obtains a list of servers from a central directory authority, and selects random relays from the list to construct a circuit for onion routing [24]. Tor requires each client to maintain a global view of all the servers. However, as the number of servers increases, maintaining a global view of the system become costly, since churn will cause frequent updates and a large bandwidth overhead. In fact, McLachan et al. [13] show that in the near future, the Tor network could be spending an order of magnitude more bandwidth in maintaining a global view of the system, than for relaying anonymous traffic.

Recent proposals to address the scalability concerns in the Tor network advocate a peer-to-peer approach [23, 15, 19, 17, 13, 20]. However, the peer-to-peer setting brings with it new challenges, including the ability to security locate random relays.

Morphmix [23] proposed to connect relays into a restricted topology and construct circuits along paths in this topology. In order to verify neighbor information returned by intermediate nodes, MorphMix designed a mechanism involving witness nodes and a collusion detection mechanism. However, the collusion detection mechanism can be circumvented by a set of colluding adversaries who model the internal state of each node, thus violating anonymity guarantees [25].

Later designs used distributed hash tables (DHT), also known as structured peer-to-peer topologies, as a foundation. Structured topologies assign neighbor relationships using a pseudorandom but deterministic mathematical formula based on the IP addresses or public keys of nodes. This allows the relationships to be verified externally, presenting fewer opportunities for attacks. Similar to Morphmix, ShadowWalker [17] also uses a random walk to locate relays, but the neighborhood information returned by intermediate nodes is verified using redundant structured topologies.

The design of Salsa [19] is similar to Tor, in that a circuit is built by selecting three random nodes in the network. Salsa uses a specially designed secure lookup operation over a custom DHT to locate forwarder nodes. The secure lookups use redundant checks to mitigate potential attacks; these checks are able to limit the bias an adversary can introduce in the lookup, but make Salsa susceptible to information leak attacks: attackers can detect a large fraction of lookups and thus infer the path structure [16]. Salsa is also vulnerable to a selective denial-of-service attack, where nodes break circuits that they cannot compromise [1, 26]. NISAN [20] and Torsk [13] are recent designs that also use DHT lookups to locate relays and include mechanisms to defend against information leak attacks; however, weaknesses in both mechanisms were later identified [27].

AP3 [15] has a similar structure where paths are built by selecting random relays using a secure lookup mechanism [3]. The design of AP3 is more similar to Crowds [22] than to Tor, with paths being formed by performing a stochastic expected-length random walk. The stochastic nature of AP3 makes it difficult for a rogue node to decide whether its preceding hop is the initiator or simply a relay in the path; however, for low-latency communication, timing attacks may make this decision simpler. Similar to Salsa, the secure lookup used in AP3 reveals a lot of information about the lookup initiator, and makes the user vulnerable to passive information leak attacks [16].

We note that all of the above approaches have focused on verification of neighborhood information returned by intermediate nodes. The mechanisms are complex, making it hard to rigorously evaluate the user anonymity, and invariably resulting in heuristic security guarantees; in fact, the security community has been very successful in breaking the state of the art P2P anonymity solutions. Finally, fingertable verification based solutions are only applicable to structured topologies, where neighbor relationships are constructed based on a deterministic pseudo-random function. They do not work with unstructured topologies like social network topologies, which are a potentially useful substrate for anonymous communication because edges between nodes represent trust relationships.

Threat Model: Low-latency anonymous communication systems are not designed to resist a global adversary. We consider a partial adversary who controls a fraction f of all the nodes in the network. This set of malicious nodes collude and can launch both passive and active attacks. We consider the set of colluding nodes is static and the adversary cannot compromise nodes at will.

Even in networks with large number of nodes, f can be a significant fraction of the network size. Powerful adversaries, such as governments or large organizations, can potentially deploy enough nodes to gain a significant fraction of the network. Similarly, botnets, whose average size has grown in excess of 20 000 nodes [21], present a real threat to anonymous systems.

3 Reciprocal Neighbor Policy

3.1 Protocol Description

We present a new primitive for scalable anonymous communication, which we call *reciprocal neighbor policy*. Our main idea is to consider bounded degree undirected versions of structured or unstructured topologies, and

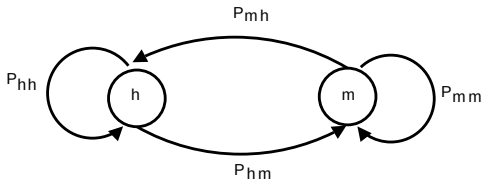

Figure 1: Attack Model.

then entangle the fingertables of neighboring nodes with each other, i.e., if a malicious node X does not advertise an honest node Y in its fingertable, then Y also excludes X from its fingertable (tit-for-tat policy). Paths for anonymous communication are constructed using a random walk [17]. The initiator first sets up a circuit with a random neighbor (finger) A, and then queries A for its fingertable. Next, the initiator selects a random node B from A's fingertable and extends the circuit to B via A. By iterating these steps, a circuit of arbitrary length can be established. We can see that if a node attempts to bias the random walk towards malicious nodes, then its own probability of getting selected as an intermediate node in the random walk reduces, nullifying the effect of the attack.

Reciprocal neighborhood policy ensures that route capture attacks on random walks serves to partially isolate malicious nodes behind a small cut in the topology, reducing the probability that they will be selected in a short random walk. Observe that our protocol requires node degrees to be bounded, otherwise the adversary can simply keep all honest nodes in its fingertable, in addition to inserting a large number of malicious nodes; thereby increasing the probability that a malicious node is selected in a random walk. We shall now show that this tit-for-tat policy is surprisingly effective at mitigating active attacks on random walks, without increasing the threat of passive attacks.

To demonstrate the effectiveness of this primitive, let us assume for now that there is a mechanism to securely achieve the reciprocal neighbor policy, i.e., that if a node X does not advertise a node Y in its fingertable, then Y also excludes X. We can model the security of the random walk process as a Markov chain. Let us suppose that there are n nodes in the network, with an average degree of d, and that a fraction f of the nodes are malicious. Also, suppose that the malicious nodes launch an active route capture attack and try to exclude y nodes each from their fingertables. We can classify the nodes into two regions, the honest set of nodes (denoted by h), and the malicious set of nodes (denoted by m). If the random walk is in the region m, then the probability of the next hop being honest is given by $P_{mh} = \frac{d-df-y}{d}$. P_{mm} can be simply computed as $1 - P_{mh}$. Similarly, if the random walk is in the region h, then the probability of the next hop being honest is given by $P_{hh} = \frac{(1-f)^2 \cdot d}{(1-f)^2 \cdot d + f \cdot (1-f) \cdot d - fy}$. Again, P_{hm} can be computed as $1 - P_{hh}$. The Markov chain with the associated transition probabilities is illustrated in Figure 1. We can now compute the probability of the l'th hop of a random walk being honest (denoted by $P(l)$) as follows:

$$P(l) = P(l-1) \cdot P_{hh} + (1 - P(l-1)) \cdot P_{mh}$$

when $l >= 1$. The terminating condition for the recursion is $P(0) = 1$, which reflects that the initiator is honest. * Figure 2 depicts the probability of l'th hop of a random walk being malicious for $n = 100\,000, d = 20, y = 10$. We can see that for small values of l, the route capture attack reduces the probability of a malicious being sampled in the random walk, but as the random walk length increases, the probability of a malicious node being sampled also increases (using the optimal attack strategy). However, even when the length of the random walk is $l = 6$, the probability of a malicious node being sampled is only slightly greater than f, using an optimal attack strategy. Note that in order to compromise user anonymity using end-to-end timing analysis, both the first and the last hops in the random walk must be malicious. When the fraction of compromised nodes is 0.2, and using an attack strategy $y = 8$ (which maximizes 6'th hop sampling bias), the probability of end-to-end timing analysis is only 0.023, which is smaller than the probability of end-to-end compromise without any route capture attack (0.04, using $y = 0$). Thus we conclude that reciprocal neighbor policy nullifies any adversarial advantage for short random walks.

3.2 Securing Reciprocal Neighbor Policy

Unstructured Topologies: In this work, we consider constant degree unstructured topologies, for example, we can consider a constant degree version of social network topologies by only admitting users who have a threshold t number of friends, and then selecting any random t edges from their friend list. Our main idea in this protocol is to have a trusted server issue one signature per time slot on each node's fingertable. This ensures that nodes can only advertise a single fingertable within the duration of any timeslot, but can update their fingertables across timeslots to handle node churn. Note that the server signature is blind and the server does not verify any information in the message. In particular, nodes are free to advertise any neighborhood information in their fingertable. The use of a trusted server ensures that there is a single fingertable advertisement per time slot. This

*For users who do not participate in the network, choosing an honest first hop is critical, as is the case with guard nodes in Tor.

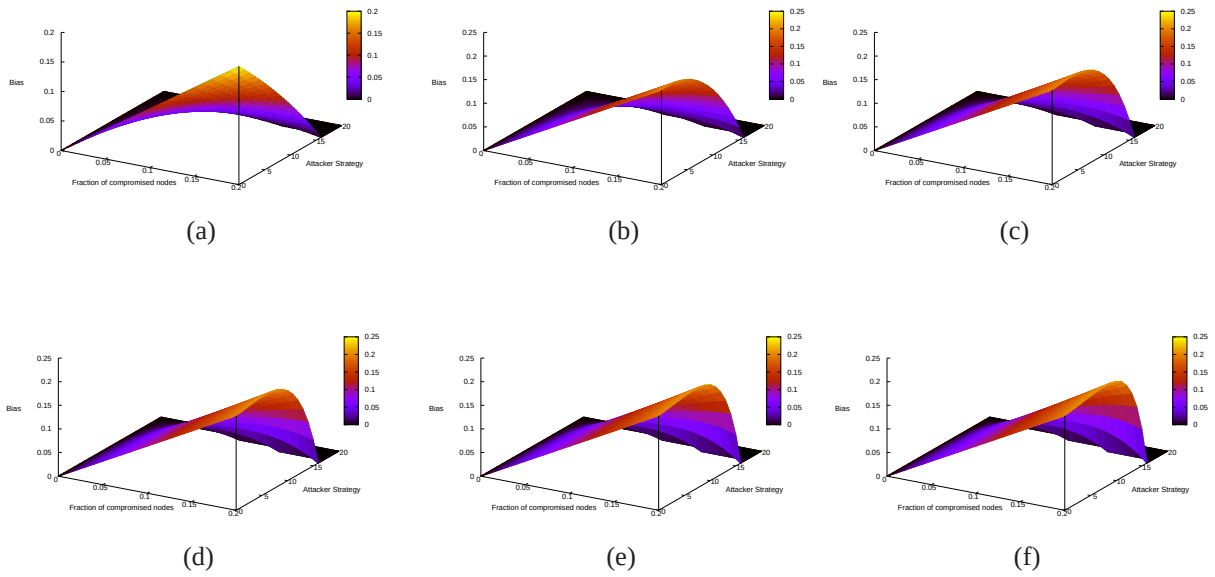

Figure 2: *Probability of l'th hop being compromised (Sampling Bias):* (a) $l = 1$, (b) $l = 2$, (c) $l = 3$, (d) $l = 4$, (e) $l = 5$, and (f) $l = 6$. Note that sampling bias is small. Moreover, attack strategies that increase sampling bias at higher random walk lengths reduce sampling bias for smaller random walk lengths. However, for end-to-end timing analysis, both first and last hops of the random walk must be compromised. Thus, reciprocal neighbor policy successfully mitigates the route capture attack.

guarantees secure reciprocal neighbor policy because a malicious node X cannot tell node Y that it is included in X's certificate, while giving another certificate that does not include Y during the random walk process. If there are n relays in the network, the trusted server will have to issue n signatures per time slot, which could be computationally expensive. We propose the following approach to further improve the scalability of our design. First, all nodes send a hash of their routing state to the trusted server. Next, the server constructs a Merkle hash tree [14] over these messages, and signs the root of the tree. Finally, the server sends its signature on the root of the tree, along with the associated $\log n$ hashes to each node, that allow a node to prove that its fingertable was part of the Merkle tree signed by the trusted server. The computational overhead of this approach is only $n \log n$ hash operations per time slot, and the bandwidth overhead is also only $O(n \log n)$ (as compared to $O(n^2)$ overhead in the current Tor architecture). Random walks on social networks are Sybil resilient [28, 5]; our solution to mitigate route capture attacks in social networks makes its use even more attractive.

Structured Topologies: While the above protocol is the *only* current proposal for mitigating route capture in unstructured topologies, it still requires the involvement of a trusted central server. Next, we propose a protocol tailored for structured topologies, where we use the structure of the topology to enforce security properties and reduce the involvement of trusted entities. Namely, we assume that there exists some sort of public key infrastructure (PKI) that is used to verifiably assign identifiers to Tor routers, but no online interaction with a trusted party is necessary.

Our main idea is to associate each node with a certificate, that is signed by its fingers. This certificate contains a list of the node's fingers. The key property that we need to impose on this certificate is that a node should be able to produce only a single certificate per time slot that is globally verifiable. The global verifiability will ensure that neighbors can check this certificate and decide whether to include it in their own fingertable, and any node in the network can also check the certificate to secure the random walk process. A single verifiable certificate per time slot ensures that a node X cannot tell node Y that it is included in X's certificate, while giving another certificate that does not include Y during the random walk process.

We use distance checking [3, 20] to ensure a certificate uniqueness per time slot. To verify a certificate, nodes compute the average distance between optimal fingers based on the structure of the topology and the fingers listed on the certificate. If this distance is greater

than a threshold, then the certificate is rejected. It has been shown in [3, 20] that distance checking has low false positives, but admits a few false negatives β, which means that an adversary will be able to replace a few honest nodes (β) in its fingertable by malicious nodes (assuming uniform distribution of node identifiers). Thus, if the distance check succeeds, then there are at most $\beta + df$ malicious nodes in the fingertable, where d is the average node degree. Consequently, the remaining $d(1-f)-\beta$ nodes are honest. In addition to the identifiers of the fingers, the certificate also consists of signatures from the fingers. We assume that honest nodes only give a single signature to a node per time slot. Thus, in order to produce two differing certificates per time slot that pass the distance checking test, the malicious node requires signatures from at least $2 \cdot (d(1-f)-\beta)$ fingers. It is easy to show that this is not possible as long as $f \leq 1 - \frac{2\beta}{d}$.

4 PIR-Tor

Securing anonymous communications peer-to-peer networks is very challenging. In order to guarantee random choice of nodes as communications relays, secure designs are built upon the assumption that the node identifiers are distributed uniformly at random in the identifier space, and that the fraction of malicious nodes in the network is less than 20%. The Tor network currently has only about 2500 relays, and if they were to be arranged in a peer-to-peer fashion today, it may not be too difficult for an adversary to compromise or operate more than 500 malicious nodes to cross the 20% threshold [18]. We propose a new client-server architecture called PIR-Tor, that is able to tolerate as many compromised relays as the current Tor network, while allowing the network to scale up to ten times its current size.

Our main insight in this architecture is that users do not need to request a complete list of available relays in the network from central directory servers; users only need a few random relays to build circuits. However, having knowledge of only a few relays in the network makes the users vulnerable to passive route fingerprinting attacks [4] from malicious directory servers even under the honest-but-curious model, which is the current threat model in Tor as well as other extensions [13]. We note that malicious servers are a real threat: recently, a Tor directory server was found to be compromised.

In PIR-Tor, we propose to leverage private information retrieval to solve the scalability problem of the Tor network. Central servers would host a database containing the IP addresses of all available Tor routers, and clients could obtain relays by querying this database using private information retrieval repeated times (as many as the length of the circuit, three in the currently deployed Tor network). This reduces the bandwidth overhead, while protecting clients privacy from adversarial directory servers. Such an approach has the added advantages of requiring minimal changes to the Tor protocol in addition to providing provable security guarantees.

In the Tor network, clients do not choose the routers fully at random but subject to several constraints. For example, the first relay must be a guard node (i.e., a stable node that has been online a significant amount of time), while the last relay must be an exit node. PIR-Tor can accommodate this requirement by adding to each entry in the database the router's policies and uptimes, but this would increase the average number of times that the database has to be queried until the IP of three suitable nodes are found. To alleviate this problem, we propose to separate the central servers in three categories, such that each category contains only IPs of guard nodes, exit nodes or middle nodes, respectively. We note that, as nodes can belong to more than one category, it could be necessary to perform some extra queries to avoid node repetitions in the paths. Nevertheless, as these databases now contain fewer nodes than in the general case explained above, the total time to obtain three IP addresses is on average smaller. Furthermore, although this strategy seems to reduce the anonymity sets of the nodes chosen by a client, it is equivalent to the current Tor mechanism as nowadays the node's role is public, i.e., it already appears in the list stored by the current directories.

A second restriction is that guard nodes are fixed for each client. This can be easily achieved in PIR-Tor, by having the client request always the same IP (i.e., the same index) from the database. Finally, relays are not chosen randomly, but in proportion to a node's bandwidth. PIR-Tor can satisfy this requirement by sorting the database in increasing order of bandwidth, and then having the client pick an index to query in a bandwidth-weighted fashion.[†]

Evaluation: We now evaluate the performance of PIR-Tor through simulations. We choose the Kushilevitz and Ostrovsky [11] computational Private Information Retrieval scheme, based on the Goldwasser and Micali homomorphic encryption scheme [9] with a key length of 1024 bits. We use the Java implementation by Bortz and Inguva [2] and run our experiments on an Intel Core2 Duo T5600 at 1.83GHz, and 3 GB of RAM.

In the current Tor network clients use three-hop paths to rely their messages. For this purpose, they regularly download a list of the currently available Tor routers from one of the central directories, and select at random three of them to construct a circuit. In PIR-Tor, clients query the server three times with random indexes, to retrieve three IP addresses. Table 1 shows the time

[†] The clients would need to learn overall distribution of bandwidth in the network, but this information need not be exact and can be updated infrequently.

Table 1: Time to obtain one IP address from the database (in milliseconds)

Nodes	1500	2500	5000	10000	15000	20000
CPIR	2.3	2.9	4.2	6.0	7.2	7.9

needed to retrieve one IP address from the server depending on the number of available routers (the numbers show the average over $100\,000$ queries and do not take into account network delays). We assume the server has a database of IP addresses, i.e., each of the entries has 32 bytes. The current Tor network counts on approximately $2\,500$ nodes. Using CPIR, a client needs 2.9 milliseconds to obtain one IP address, thus only 9 milliseconds are necessary to obtain three routers to form a circuit. Even if the network grows up to 20 000 nodes only 7.9 milliseconds per query are needed (24 milliseconds to construct a path). Thus, a single server can support $25,000$ clients building a single circuit every 10 minutes. Further improvements in scalability can be achieved by replicating the server.

Another promising approach to implement PIR-Tor is using information theoretic PIR [8]. In this protocol multiple servers hold a copy of the database, and the client queries all of them to retrieve information. A t-private l-server PIR is a PIR system in which the privacy of the query is information theoretically protected, even if up to t of the l servers collude.

5 Conclusion

Tor [7] requires users to select routers at random in order to provide anonymity. For this purpose users must maintain a global view of the system. It has been shown that, if the Tor network continues growing at the same rate [12] this approach will be soon impracticable [13].

In this paper we have proposed two alternative architectures that solve this problem. The first one is a peer-to-peer architecture, in which the appearance of nodes in each other fingertables is reciprocal, and we demonstrate how this mechanism allows for better random sampling of nodes than previous proposals, substantially reducing the probability of route capture.

Our second contribution is PIR-Tor, a centralized architecture that leverages Private Information Retrieval to suppress the need for global knowledge of the system. Although this solution trades off bandwidth for computation, we prove through simulations that the latter is within reach of off-the-shelf computers.

References

[1] BORISOV, N., DANEZIS, G., MITTAL, P., AND TABRIZ, P. Denial of service or denial of security? *ACM CCS* (2007).

[2] BORTZ, A., AND INGUVA, S. pir0.1, 2005.

[3] CASTRO, M., DRUSCHEL, P., GANESH, A., ROWSTRON, A., AND WALLACH, D. S. Secure routing for structured peer-to-peer overlay networks. In *OSDI* (December 2002).

[4] DANEZIS, G., AND CLAYTON, R. Route Fingerprinting in Anonymous Communications. *IEEE P2P* (2006).

[5] DANEZIS, G., AND MITTAL, P. Sybilinfer: Detecting sybil nodes using social networks. In *NDSS* (2009).

[6] DANEZIS, G., AND SYVERSON, P. Bridging and fingerprinting: Epistemic attacks on route selection. In *PETS* (2008).

[7] DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. Tor: The second-generation onion router. In *USENIX Security* (2004).

[8] GOLDBERG, I. Improving the robustness of private information retrieval. In *IEEE S&P* (2007), IEEE Computer Society.

[9] GOLDWASSER, S., AND MICALI, S. Probabilistic encryption. *J. Comput. Syst. Sci. 28*, 2 (1984), 270–299.

[10] GOODIN, D. Tor at heart of embassy passwords leak. *The Register* (September 10 2007).

[11] KUSHILEVITZ, E., AND OSTROVSKY, R. Replication is not needed: single database, computationally-private information retrieval. In *FOCS 97*.

[12] LOESING, K. Measuring the tor network: Evaluation of client requests to the directories. *Tech. Report* (2009). `https://git.torproject.org/checkout/metrics/master/report/dirreq/directory-requests-2009-06-26.pdf`.

[13] MCLACHLAN, J., TRAN, A., HOPPER, N., AND KIM, Y. Scalable onion routing with torsk. *ACM CCS* (November 2009).

[14] MERKLE, R. Protocols for public key cryptosystems. In *IEEE S&P* (1980).

[15] MISLOVE, A., OBEROI, G., POST, A., REIS, C., DRUSCHEL, P., AND WALLACH, D. S. Ap3: Cooperative, decentrialized anonymous communication. *ACM SIGOPS European Workshop* (2004).

[16] MITTAL, P., AND BORISOV, N. Infomation leaks in structured peer-to-peer anonymous communication systems. *ACM CCS* (2008).

[17] MITTAL, P., AND BORISOV, N. Shadowwalker: Peer-to-peer anonymous communication using redundant structured topologies. *ACM CCS* (2009).

[18] MURDOCH, S. J., AND WATSON, R. N. M. Metrics for security and performance in low-latency anonymity systems. In *PETS* (2008).

[19] NAMBIAR, A., AND WRIGHT, M. Salsa: A structured approach to large-scale anonymity. *ACM CCS* (2006).

[20] PANCHENKO, A., RICHTER, S., AND RACHE, A. Nisan: Network information service for anonymization networks. *ACM CCS* (November 2009).

[21] PAULSON, L. D. News briefs. *IEEE Computer 39(4):17-19* (April 2006).

[22] REITER, M., AND RUBIN, A. Crowds: Anonymity for web transactions. *ACM TISSEC 1*, 1 (June 1998).

[23] RENNHARD, M., AND PLATTNER, B. Introducing morphmix: Peer-to-peer based anonymous internet usage with collusion detection. *WPES* (2002).

[24] SYVERSON, P., TSUDIK, G., REED, M., AND LANDWEHR, C. Towards an analysis of onion routing security. *Workshop on Design Issues in Anonymity and Unobservaility, vol. 2009, LNCS, Springer* (July 2000).

[25] TABRIZ, P., AND BORISOV, N. Breaking the collusion detection mechanism of MorphMix. In *PET* (2006).

[26] TRAN, A., HOPPER, N., AND KIM, Y. Hashing it out in public: common failure modes of dht-based anonymity schemes. In *WPES* (2009).

[27] WANG, Q., MITTAL, P., AND BORISOV, N. In search of an anonymous and secure lookup. In *ACM CCS* (2010).

[28] YU, H., GIBBONS, P. B., KAMINSKY, M., AND XIAO, F. Sybillimit: A near-optimal social network defense against sybil attacks. In *IEEE S&P* (2008).