Collaborative Red Teaming for Anonymity System Evaluation

Sandy Clark° Chris Wacek• Matt Blaze°
Boon Thau Loo° Micah Sherr• Clay Shields• Jonathan Smith°

°*University of Pennsylvania* •*Georgetown University*

Abstract

This paper describes our experiences as researchers and developers during red teaming exercises of the SAFEST anonymity system. We argue that properly evaluating an anonymity system — particularly one that makes use of topological information and diverse relay selection strategies, as does SAFEST— presents unique challenges that are not addressed using traditional red teaming techniques. We present our efforts towards meeting these challenges, and discuss the advantages of a *collaborative* red teaming paradigm in which developers play a supporting role during the evaluation process.

1 Introduction

The need for secure, dependable, and high performance systems to enable anonymous communication in the presence of eavesdroppers has been the focus of much research in the past several years (cf. the survey by Sampigethay and Poovendran [15]). Perhaps the most successful of these efforts is the volunteer-operated Tor network [7], recently estimated to enhance the privacy of as many as hundreds of thousands of daily users [9].

Unfortunately, as has been pointed out by the network's operators [8], Tor suffers from significant congestion, leading to latencies and bandwidths that are an order of magnitude worse than those offered by unprotected direct communication. As has been noted in previous work, Tor's performance problems are due in part to the asymmetry between the number of clients and anonymizing relays [8], the presence of bandwidth-intensive filesharers on the network [13], and the use of latency-agnostic relay selection strategies [2, 17, 18].

In this paper, *we describe our experiences as developers during recent red teaming efforts* to test and analyze Selectable Anonymity for Enabling SAFER Telecommunications (SAFEST) – a tunable and latency-aware enhancement to Tor that attempts to alleviate this latter cause of Tor's slowness. SAFEST extends existing work on link-based routing [17, 18] in which clients (sometimes called *initiators*) weigh relay selection based on the expected end-to-end (e2e) cost of *link* performance indicators such as latency, AS hop count, and jitter.

SAFEST differs from existing anonymity systems in its ability to establish reliable channels with predictable QoS characteristics. In contrast to existing anonymity systems which rely on immutable relay selection algorithms, SAFEST allows the sender to provide a relay selection policy that precisely defines the manner in which relays are chosen for anonymous paths. This effectively lets the initiator control the e2e performance of her anonymous paths, since a path's performance is dictated in large part by the network conditions and available resources at its constituent segments. Built on top of the existing Tor platform, SAFEST leverages distributed Internet embedding systems [5] to provide compact encodings of network state. Using these embedding systems, SAFEST provides a mechanism for initiators to make *informed* choices when selecting relays. This flexibility permits more fine-tuned relay selection and better conforms to applications' specific performance requirements (e.g., latency, bandwidth, jitter, etc.).

SAFEST's features — in particular, its use of network topology information to construct better performing anonymous paths — present a number of interesting challenges for experimentation and evaluation. Since SAFEST permits a variety of relay selection strategies whose performance depends on dynamic network conditions, accurately modeling realistic deployments becomes critically important for evaluating the system's effectiveness. Drawing on our experiences as SAFEST developers and as observers of its red teaming evaluation,

the focus of this paper is on (1) the challenges inherent in analyzing the security properties of an anonymity system and (2) best practices that may be applied to meet these challenges.

Contributions. We present our initial efforts towards mitigating the challenges associated with evaluating a system such as SAFEST. In particular, we describe the development of a clonable and scalable network topology that accurately models network characteristics as well as client behavior.

We additionally describe our experiences, from our vantage point as "Blue Team" researchers and developers, participating in *interactive* red teaming exercises. During the red team evaluation, we operated in concert with an independent and external team of "attackers" to discover possible weaknesses in our design and implementation. We argue that this collaborative model in which the Red Team does not act in strict isolation is crucial for framing realistic attack scenarios and analyzing a complex and topology-aware anonymity system.

Finally, we describe how our combined Red Team/Blue Team exercises led to the discovery of a new variation in a class of attacks on Internet embedding systems that may not otherwise have been found by the Red Team working alone.

We begin by describing in more detail the unique challenges of evaluating a performance-driven anonymity system.

2 Challenges

Measuring the performance and anonymity characteristics of an Internet-based anonymity system presents a number of interesting research challenges. We highlight some of these challenges below.

Challenge 1: Utilizing an expressive measurement framework. We require an effective measurement framework that enables controlled testing of the system. This framework should provide a number of capabilities:

- **Configurable:** The design of the network and the nodes within it can be precisely specified with as few constraints as possible;
- **Controlled:** Inputs are limited to those specified by the experimenter. This isolation is necessary to allow developers and testers to better reason about the system's behavior;
- **Measurable:** Traces, logs, and other data can be obtained from any point of the network; and
- **Reproducible:** An experiment can be run multiple times and produce similar results.

Configurability and controllability enable a researcher to rapidly iterate through experiments, validating different attack vectors or tweaking system parameters. At the same time, controllability, measurability, and reproducibility allow data from separate experiments to be comparable.

Perhaps counterintuitively, we argue that live deployments are not well-suited for regularly conducting anonymity experiments. Although they offer maximum realism, live networks are difficult to configure, making it difficult to generalize the results of any particular experiment beyond the tested network. Moreover, a live anonymity network does not exist in isolation and may be affected by uncontrollable and unpredictable events that occur on the Internet (e.g., routing changes, traffic bursts, etc.), making it more difficult to reproduce experimental results. Finally, experimenting on live anonymity networks may expose its users to risk [3, 19], since experimental changes may detrimentally affect user's anonymity. There are no currently widely accepted community standards for conducting research on live anonymity networks [19].

Challenge 2: Constructing realistic topologies. The performance of any network overlay is partially dependent on the configuration of the network on which it is deployed. In particular, an anonymity network's throughput is a function of the bandwidth available at its anonymizing relays; e2e path latencies are comprised in part from the latencies between relays; and jitter on the overlay results from both transient network effects as well as changing workloads on the network's nodes. Studies that rely on overly simplistic topologies risk missing effects that may produce degraded performance or anonymity under more realistic conditions. Even for anonymity systems in which performance is less critical, the failure to consider network characteristics may cause researchers to overlook attacks that leverage network effects (e.g., the loss or delay of a particular message).

Carefully constructing topologies is necessary to understand (1) how an anonymity protocol will function under various network conditions and (2) how attackers can exploit topological information to decrease participants' anonymity. This is particularly critical for evaluating "network-aware" anonymity systems such as SAFEST where relay selection is fully dependent on the *perceived* network topology.

Challenge 3: Accurately modeling client behavior. Accurately modeling client behavior is especially chal-

lenging when experimenting with anonymity systems. A correct anonymity system will, by design, thwart attempts at understanding the (potentially highly variant) behavior of its users. Hence, previous efforts at quantifying client behavior on live networks rely on statistical sampling [13, 20]. Although such approaches may be used to approximate user behavior, they are inherently biased toward *existing* behavior on current networks rather than on *desired* usage. For example, although the latencies on Tor are too high to permit real-time video communication, users of the anonymity service may appreciate such a capability and might participate in anonymous video conferencing if it were possible.

Enumerating and characterizing client behavior is particularly challenging for SAFEST. Since the anonymity system enables clients to choose diverse relay selection strategies, a significant challenge during SAFEST's red teaming exercises was determining appropriate client behavior according to a variety of communication requirements.

Challenge 4: Accurately modeling adversarial behavior. There is a large body of work that considers attacks against anonymity systems (cf. the survey by Sampigethaya and Poovendran [15]). Attackers can be classified both according to their abilities — e.g., administrators of large networks, botmasters, individual rogue operators, etc. — as well as their motives — e.g., to break anonymity, to disrupt the network, and (for performance-centric anonymity services such as SAFEST) to degrade performance. Moreover, compared to closed distributed systems with more stringent access controls, anonymity systems that rely on volunteer-operated relays have a comparatively large attack surface: attackers can be either external to the network, or may operate as malicious insiders (e.g., relay operators). A significant challenge of red-teaming an anonymity system is thus to model realistic adversaries and quantify the risk they pose to the network.

3 Methodology

In what follows, we present our experiences before and during the red teaming SAFEST exercises, and in particular, describe the methods we applied as developers to meet the above challenges.

3.1 Experimental Framework

Recently, the ExperimenTor [3] and Shadow [11] frameworks have been proposed for anonymity system experimentation. Both frameworks are designed for measuring performance characteristics on large virtual deployments, and support the execution of unmodified (or slightly modified) binaries on top of emulated [3] or simulated [11] network topologies.

Framework	Advantage(s)	Disadvantage(s)
DETER	Actual hardware, on-the-fly configuration changes	Limited resources and contention
ExperimenTor	Unmodified binary emulation, real time experiments	Limited scalability
Shadow	Low hardware costs, scalable	Modified binaries, slower than real time

Table 1: Experimental frameworks for Internet testing. We relied on ExperimenTor for SAFEST development and DETER for the red teaming exercises.

For the SAFEST exercises, we utilized ExperimenTor for system development and the DETER testbed [6, 14] for the red teaming exercises. The DETER testbed is a public facility for medium-scale repeatable experiments in computer security. Built using the University of Utah's Emulab software, the DETER testbed has been configured and extended to provide effective containment for a variety of computer security experiments. In our setting, DETER allows us to run SAFEST code on reserved (i.e., not shared) machines and reconfigure the topology according to our needs.

Table 1 presents a high-level comparison of experimental frameworks for testing anonymity systems. A more thorough discussion of the advantages and disadvantages of various experimental frameworks is beyond the scope of this paper; interested readers may refer to the arguments motivating the development of ExperimenTor and Shadow [3, 11].

3.2 Topology Generation

Accurate testing depends upon a network topology that captures certain characteristics of the real world Internet: it must have a diversity of link latencies and link bandwidths, as well as a sufficient number of nodes to instantiate a diverse set of paths within the overlay network.

In our testing of SAFEST on DETER, we made use of a three tier "core-stub-node" topology. Our topology, depicted in Figure 1, is a three level tree in which all application nodes operate at the leaves of the tree, the 'core' is the root of the tree, and 'stubs' are branches. The root of our experimental topology consisted of three core root nodes, each of which connects to three stub nodes. Each stub contains five application nodes. The resulting topol-

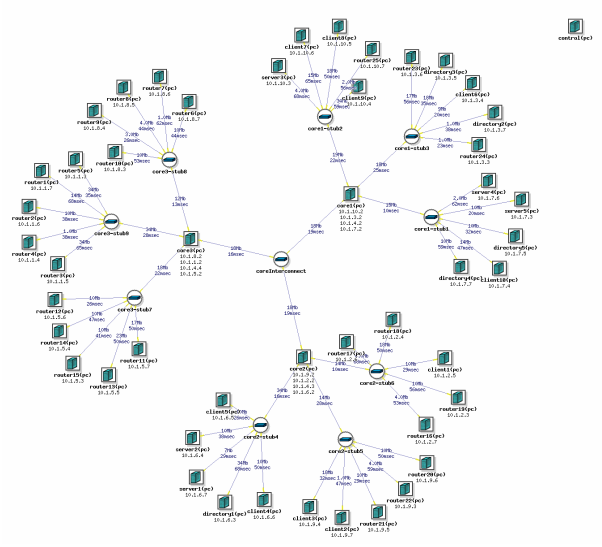

Figure 1: Experiment topology on DETER. The disconnected node in the top-right corner is used to control the experiments and does not participate in the anonymity network.

ogy contains 45 application nodes, providing a wide variety of possible paths through the overlay network while minimizing the hardware required.

To create a diversity of link latencies and bandwidths, we randomly select bandwidths and latencies from a distribution that approximates the variety of latencies and bandwidths present on the Internet. Core and core-stub connections are high bandwidth and low latency to simulate the Internet backbone, while leaf connections are lower bandwidth and higher latency to simulate last-mile connections. Core latencies were chosen uniformly at random from $[10, 18]$ ms; leaf latencies were selected uniformly at random from $[20, 68]$ ms. The core bandwidths were sampled from a distribution that cross references the geographic distribution of Tor clients in early 2011 with bandwidth rates from those countries. The bandwidth data were obtained from NetIndex.com, which aggregates SpeedTest.net data[1].

These sample distributions, while not exact representations of real-world bandwidths and latencies, provide a varied performance experience for nodes within the topology, and permits anonymous paths with diverse performance characteristics.

Utilizing dual-frameworks. We deployed two duplicate instances of the experimental configuration on the DETER testbed. The redundancy in the experimental setups provides independent *control* and *experiment* settings for better measuring the effects of configuration or protocol changes. In a time-constrained evaluation session, the dual setups have the additional advantage of providing a failover instance if aggressive experimentation results in the primary experimental instance becoming non-functional.

[1]Speedtest.net [`http://speedtest.net`] is a service that permits end users to identify the speed of their network connections by downloading files from known 'fast' servers and measuring the throughput.

3.3 Client Behavior

Many different types of traffic flow through the Tor network today. The vast majority is standard HTTP web traffic [13]. Despite the fact that HTTP traffic comprises a significant fraction of Tor traffic, additional types of traffic should be considered when evaluating an anonymity system such as Tor. For instance, modifications which exclusively aim to enhance the performance of one particular class of applications may have an adverse effect on other application classes.

When evaluating SAFEST, we sought to model a variety of potential uses for a performance-driven anonymity system. Our clients engaged in several behaviors intended to capture the performance characteristics associated with these uses.

First, each client uses `curl` to request files of various sizes from a remote web server. We use a range of file sizes intended to capture regular web browsing activity, where most files are under 500K; we additionally model "bulk" downloaders who regularly transfer large files.

Second, clients periodically engage in fixed bit-rate communication with a destination server for a randomly-selected period of time. This is designed to capture the characteristics important to low-latency applications (in particular, VoIP clients).

3.4 Evaluation Methodology

The need for red teaming a security application is well understood, and there has been much work to develop standards for assessing information security [10, 16]. As early as 1973 Clark Weissman [21] developed a 'Flaw Hypothesis Methodology' which delineated the steps involved in a successful penetration test, and this methodology is still used in some form today. Two early studies [1, 12] explored the benefits of penetration testing systems security using this methodology. However, both studies viewed the 'testing phase' of penetration testing as best done as a "Gedanken" or thought experiment, rather than something performed by active red team attackers.

Today red teaming usually takes one of two forms: *overt* and *covert*. Overt testing involves internal and/or external evaluation of a system with the knowledge and support of the system's operators. In contrast, covert testing occurs without the knowledge of the developers or IT staff. Traditionally, neither overt nor covert testing is performed with the active participation or cooperation of system developers.

The majority of literature concerning security testing focuses on red teaming a particular instance of a system, network, or corporate entity, rather than on testing the design of a system. For example, in recommending whether to choose overt or covert testing, NIST states:

> Overt testing is less expensive, carries less risk than covert testing, and is more frequently used– but covert testing provides a better indication of the everyday security of the target organization because system administrators will not have heightened awareness. [16]

Importantly, this distinction does not apply to testing a design, since a design, unlike an instance, has no system administrators. When evaluating the design of a system — as is the case in the SAFEST exercises — there is little benefit for the red teaming to be covert.

While the attack vectors are slightly different when focusing on a distributed anonymity system, a successful evaluation and red team exercise for an anonymity system retains many of the elements from standard red team exercises. Below, we describe the components of the SAFEST red teaming exercises, including the addition of the *Sandboxing* phase that emphasizes the non-adversarial relationship between the system developers and the Red Team.

- *Scope Setting and Brainstorming*: System designers and Red Team members meet prior to the evaluation exercises to identify attack surfaces that are in scope and discuss potential attack vectors. This aids the Red Team, who may be unfamiliar with the particulars of a new anonymity system.
- *Independent Vulnerability Analysis*: The Red Team analyzes the applicable attack surface of the system, documents potential vulnerabilities, and develops a plan of attack for evaluating those vulnerabilities. (Vulnerabilities in the baseline Tor software were treated as "out-of-scope" for the purpose of this red teaming exercise since any released Tor patches could be applied to the SAFEST codebase.)
- *Exploitation*: During the evaluation period, the Red Team attempts to independently implement the attacks that they have developed against the system.
- *Sandboxing*: During the evaluation period, post-Exploitation, the Red Team and co-located system developers discuss the vulnerabilities discovered by the Red Team. During this stage, system developers and Red Team members discuss and tweak parameters in the running experimental instance to better characterize the vulnerabilities and measure their severity. As we argue in what follows, the intersection of Red Team focus and system developer expertise can perceive more specific issues than either independently.
- *Risk Analysis and Quantification*: Red Team members quantify their assessment of the risks posed by the identified vulnerabilities. These assessments are validated by the system developers.

This enhanced evaluation methodology can provide greater clarity into the vulnerabilities exposed by a system, while the system developer involvement in the process engenders greater ownership of the identified issues.

Critically, the organization overseeing the development and evaluation effort must reinforce that identified vulnerabilities do not affect project performance, or evaluation processes can swiftly turn antagonistic.

4 The School of Fish Attack: An Example of Collaborative Red Teaming

The Red Team identified several previously unknown vulnerabilities in the tested SAFEST implementation. In what follows, we describe one such exploit — the *School of Fish* attack — and discuss how the attack's discovery was made possible due to the interactions between the system's designers and its evaluators. We emphasize that the main contribution of this paper is not the discovery of the School of Fish attack, but rather the argument that collaborative red teaming offers advantages over more traditional and isolated red teaming approaches. The example below is intended to highlight these advantages.

Background. To provide low latency anonymous circuits, SAFEST makes use of a virtual coordinate embedding system. Each relay is assigned an n-dimensional virtual coordinate such that the Euclidean distance between any two relays' coordinates should serve as a useful estimate of the latency between the pair. SAFEST currently uses a slightly modified version of the Vivaldi virtual coordinate system [5]: each relay selects another relay at random, requests its coordinate, and empirically measures the roundtrip time between them; the requesting relay then adjusts its coordinate to reduce the difference between the virtual and empirical distances.

Since SAFEST uses virtual distances to estimate the costs of routing between relays, an adversary who games

Figure 2: A malicious relay (red circle) joins the network and is located next to a number of targeted honest relays (grey circles).

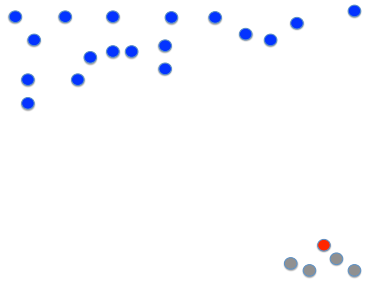

Figure 3: The malicious relay slowly adjusts its coordinates, causing a "School of Fish" effect in which the victims (grey nodes) follow it to an undesirable location in the coordinate space.

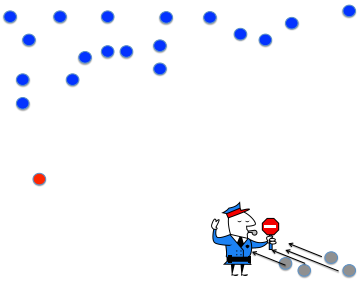

Figure 4: Even after the malicious relay stops its attacks, the SAFEST security mechanism (police officer) prevents the honest relays from migrating back to their original positions.

the coordinate system can harm anonymity by causing malicious relays to appear more attractive (or conversely, non-malicious relays to appear less attractive). To mitigate such attacks, SAFEST implements a simple security mechanism: before a relay accepts an advertised coordinate from a peer, it assesses whether updating its own coordinate based on the advertised coordinate will increase or decrease its *error* using the most recently observed coordinates and empirical measurements. Here, error is defined as the median difference between the virtual $(v_1, \ldots, v_k)$ and actual $(d_1, \ldots, d_k)$ distances between itself and k of its neighbors.
That is,

$$\text{error} = \text{median}(\bigcup_{i=1}^{k} |v_i - d_i|)$$

If the error increases by more than a threshold amount, the requesting relay does not update its coordinate. Conceptually, the security measure ensures that an advertised coordinate will not too adversely influence a relay's own coordinate.

Discovering the School of Fish attack. During the red teaming exercises, the Red Team attempted to isolate a targeted honest relay to make it appear less attractive. The attack consisted of using malicious relays to advertise carefully selected but incorrect virtual coordinates in order to "push" the targeted relay to a far off region of the coordinate space. Due to SAFEST's coordinate protection mechanism, the attack was unsuccessful.

Importantly, however, the presence and participation of SAFEST developers *during* the red teaming exercises helped lead to the discovery of a novel attack. When the above described attack failed, *the developers were able to explain why it failed* during the Sandboxing process. In collaboration with the Red Team, the developers were able to reason about the behavior of the system during the tested attack.

The interaction between the two teams aided the Red Team in refining their attack. In particular, with input from the developers, the Red Team discovered that our simple coordinate security check could be exploited to *worsen* an attack against the coordinate system, under certain network conditions. By introducing a "choke point" in the network — a capability available to a moderately advanced attacker — the adversary was able to cause a *School of Fish* effect in which many honest relays in a network partition slowly followed a malicious relay to far off regions of the coordinate space (Figures 2 and 3). After the adversary halted the attack and the network became unpartitioned, the coordinate protection mechanism prevented the system from re-stabilizing for a period of time (Figure 4) since the affected relays did not trust the now far-away coordinates offered by their unaffected (but still honest) peer relays.

Our current research efforts explore strategies to defeat this newly identified attack, including the use of recently proposed coordinate protection schemes [4].

5 Lessons Learned

Successfully evaluating a distributed network anonymity system is a difficult task. The evaluation framework, the tested topologies, and the behavior of the clients, relays, and adversaries must be carefully configured in order to thoroughly exercise the elements of the anonymity system. These challenges are especially pronounced for network-aware anonymity systems such as SAFEST

where the behavior of the system depends on the perceived network environment.

The SAFEST red teaming exercises were successful because we used an experimental framework (DETER) that allows on-the-fly adjustments and effective measurement. We developed a topology and client behavior model that tested the elements of the SAFEST system. And perhaps most importantly, the collaborative organization of the red teaming exercises and the high level of interaction between the system developers and the Red Team enabled the discovery of a vulnerability *located within a security mechanism* that might not otherwise have been detected.

From our experiences as developers during the SAFEST red teaming process, we can extract several lessons from our methodology that may be applied to other red teaming exercises:

Lesson 1: Developer participation in red teaming should not be verboten. Conventional red teaming exercises are usually carried out without developers' participation, and sometimes without their knowledge [16]. There is obvious and well-established value in maintaining *independence* between system designers and evaluators: the persons testing a system should not abide by the same set of assumptions and prejudices as the system's developers.

However, we argue that testers can benefit from the insights and experiences that may be offered by the system's designers. We highlighted one such example in the case of SAFEST in which interactions between developers and evaluators led to the discovery of a novel attack. But more generally, there is a definite benefit to gaining feedback of attacks (both successful and unsuccessful) from developers (i.e., the "sandboxing" process described in Section 3.4). Given the time constraints imposed on any realistic red teaming scenario, the behavior of complex and oftentimes distributed systems is best explained by the system's architects. The availability of this knowledge only benefits the red team.

Lesson 2: The network should be considered an attack surface. The attack surface of an anonymity system differs somewhat from that of a standard computer system. Red teams and penetration exercises are typically focused on gaining unauthorized access and subverting intended behavior. However, in a distributed anonymity system such as SAFEST, compromising a single element of the network may be insufficient to break anonymity, and common attack vectors such as social engineering and phishing to obtain passwords have little meaning.

The focus of red teaming exercises against distributed anonymity systems should not exclude the ability to subvert the network, since sufficient vulnerabilities may compromise the system in the aggregate (for example, the ability to partition the network permits the discovered School of Fish attack). Red teams should pay particular focus on the areas where the behavior of an attacker may compromise anonymity *without* necessarily assuming complete control.

Lesson 3: A successful *collaborative* red teaming exercise requires careful organization. The advantages of having a third-party evaluate the security of a system depend on the ability of the evaluators to freely assess the system without biases from the developers. In a collaborative red team exercise, this independence should be strictly maintained: the red team should operate according to their own plans and without influence from developers. During the red teaming process, the developers should maintain only a supporting role, answering questions and helping to explain system behavior.

The relationship between the red team and the developers should not be adversarial. In particular, it should be the common goal of each group to identify (and possibly resolve) vulnerabilities. To enable participants to communicate freely without the need for self-censorship, there should be no repercussions to the developers for detected vulnerabilities.

Lesson 4: Physical presence matters. During SAFEST testing, the Red Team and SAFEST developers were *co-located* for the duration of the evaluation stage. This made possible the sandboxing phase of the evaluation methodology and likely reduced any confusion that would have been due to high latency communication. We advocate that future red teaming exercises adopt this model. Onsite access to developers enables more immediate and unfettered collaborations, and promotes the use of developers as available resources during the red teaming process.

6 Conclusion

This paper recounts our experiences as SAFEST developers during the system's red teaming exercises. We argue that a network-aware anonymity system such as SAFEST presents new and interesting challenges for red teaming, and that a useful tool for meeting these challenges is the inclusion of system developers and designers in the red teaming process.

Acknowledgments

We thank our shepherd, Stephen Schwab, and the anonymous reviewers for their comments and suggestions. We are particularly grateful to Brian Caswell and the other members of the Raytheon SI red team for their expertise and tutelage, and for allowing us to participate in the red teaming process. We would also like to express our thanks to the members of the DETER team for helping us design, model, and test our topology.

This work is partially supported by NFS CAREER CNS-1149832. This material is based upon work supported by the Defense Advanced Research Project Agency (DARPA) and Space and Naval Warfare Systems Center Pacific under Contract No. N66001-11-C-4020. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Project Agency and Space and Naval Warfare Systems Center Pacific.

References

[1] M. D. Abrams, S. G. Jajodia, and H. J. Podell, editors. *Information Security: An Integrated Collection of Essays*. IEEE Computer Society Press, Los Alamitos, CA, USA, 1st edition, 1995.

[2] M. Akhoondi, C. Yu, and H. V. Madhyastha. LASTor: A Low-Latency AS-Aware Tor Client. In *IEEE Symposium on Security and Privacy (Oakland)*, 2012.

[3] K. Bauer, M. Sherr, D. McCoy, and D. Grunwald. ExperimenTor: A Testbed for Safe and Realistic Tor Experimentation. In *USENIX Workshop on Cyber Security Experimentation and Test (CSET)*, 2011.

[4] S. Becker, J. Seibert, C. Nita-Rotaru, and R. State. Securing Application-Level Topology Estimation Networks: Facing the Frog-Boiling Attack. In *International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2011.

[5] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. *SIGCOMM Comput. Commun. Rev.*, 34(4):15–26, 2004.

[6] DETER Network Security Testbed. http://www.isi.deterlab.net/.

[7] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium (USENIX)*, 2004.

[8] R. Dingledine and S. Murdoch. Performance Improvements on Tor, or, Why Tor is Slow and What We're Going to Do About It. https://svn.torproject.org/svn/projects/roadmaps/2009-03-11-performance.pdf, March 2009.

[9] S. Hahn and K. Loesing. Privacy-preserving Ways to Estimate the Number of Tor Users, November 2010. Available at https://metrics.torproject.org/papers/countingusers-2010-11-30.pdf.

[10] P. Herzog. Open-Source Security Testing Methodology Manual 2.1. Special Publication 2.1, Institute for Security and Open Methodologies, August 2003.

[11] R. Jansen and N. Hopper. Shadow: Running Tor in a Box for Accurate and Efficient Experimentation. In *Network and Distributed System Security Symposium (NDSS)*, 2012.

[12] R. R. Linde. Operating system penetration. In *Proceedings of the May 19-22, 1975, national computer conference and exposition*, AFIPS '75, pages 361–368, New York, NY, USA, 1975. ACM.

[13] D. McCoy, K. Bauer, D. Grunwald, T. Kohno, and D. Sicker. Shining Light in Dark Places: Understanding the Tor Network. In *Privacy Enhancing Technologies Symposium (PETS)*, 2008.

[14] J. Mirkovic, T. Benzel, T. Faber, R. Braden, J. Wroclawski, and S. Schwab. The DETER Project: Advancing the Science of Cyber Security Experimentation and Test. In *IEEE International Conference on Technologies for Homeland Security (HST)*, 2010.

[15] K. Sampigethaya and R. Poovendran. A Survey on Mix Networks and Their Secure Applications. *Proceedings of the IEEE*, 94(12):2142–2181, December 2006.

[16] K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh. Technical Guide to Information Security Testing and Assessment. Special Publication 800-115, National Institute of Standards and Technology, September 2008.

[17] M. Sherr, M. Blaze, and B. T. Loo. Scalable Link-Based Relay Selection for Anonymous Routing. In *Privacy Enhancing Technologies Symposium (PETS)*, August 2009.

[18] M. Sherr, A. Mao, W. R. Marczak, W. Zhou, B. T. Loo, and M. Blaze. A3: An Extensible Platform for Application-Aware Anonymity. In *Network and Distributed System Security Symposium (NDSS)*, 2010.

[19] C. Soghoian. Enforced Community Standards For Research on Users of the Tor Anonymity Network. In *Workshop on Ethics in Computer Security Research (WECSR)*, 2011.

[20] Tor Project, Inc. Tor Metrics Portal. https://metrics.torproject.org/.

[21] C. Weissman. System Security Analysis/Certification Methodology and Results. Technical report, October 1973.