Simulating Malicious Insiders in Real Host-Monitored User Data

Kurt Wallnau, Brian Lindauer, Michael Theis
Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA.

Robert Durst, Terrence Champion, Eric Renouf, Christian Petersen
Skaion Corp., N. Chelmsford, MA.

Abstract

Our task is to produce test data for a research program developing a new generation of insider threat detection technologies. Test data is created by injecting fictional malicious activity into a background of real user activity. We rely on fictional narratives to specify threats that simulate realistic social complexity, with "drama as data" as a central organizing metaphor. Test cases are scripted as episodes of a fictional crime series, and compiled into time-series data of fictional characters. Users are selected from background to perform the role of fictional characters that best match their real-world roles and activities. Fictional activity is *blended* into the activity of real users in the cast. The cast and unmodified background users perform dramas in test windows: performances are test cases. Performances by different casts of users, or by the same cast of users in different test windows, constitute distinct test cases.

Keywords

insider threat; synthetic test data

1 Introduction

Obtaining human subject test data with sufficient scale and realism to test cyber-security analytics is known to be challenging. Test data (we henceforth assume "human subject") usually consists of some mix of synthetic and real data. These two kinds of data have respective strengths and limitations that make them suitable for use in different situations. Synthetic data has the merit of being available (in principle) at low cost and to arbitrary scale. However, the correspondence of simulated and real data can be established for only a few dimensions at a time; testing beyond these select regions will not produce reliable results [3, 4]. In contrast, real data has the merit of realism *non plus ultra*, but is often difficult to obtain, and if obtained, is likely to be unlabeled [10].

However, even fully labeled real data will be inadequate as test data for *low and slow* threat domains, i.e., where malicious user activity is a vanishingly small fraction of all user activity ("low"), and threats unfold over weeks and months ("slow"). Insider threat is one such domain if we assume a .02% base rate for malicious insiders in the workforce (low),[1] and in addition include precursors to overt acts in the threat domain (slow). In low and slow threat domains, reality presents an insufficient test sample. A dataset representing the activity of $5,000$ business users (as reported in this paper) would contain (on average) only one malicious insider, and we would be exceedingly "lucky" if overt malicious acts would appear in the time windows presented by the data. Put bluntly, the question confronting insider threat research is not *whether* but *how* to construct synthetic test data.

We describe a technique that leverages the "realism" of real background data to produce synthetic insider threat test data that is rich in social complexity, plausible and above all realistic, while simultaneously avoiding artifacts. We use background data in two ways: as a source of actors to simulate dramatic performances of threat dramas in background ; and as deep *fabula* that captures real life detail that is far beyond the reach of traditional narrative. We describe our technique, its motivation, and the results we have producing red team threat data for a DARPA-sponsored insider threat research program.

2 Related Work

Sommer and Paxon [9] provide a definitive account of the methodological challenges of developing and testing machine learning-based network intrusion detectors. Many issues they describe—notably, lack of ground truth, behavioral variability, semantic gap—are present in our work, in many cases amplified by the complexity of social behavior. Aviv and Haeberlen [1] report similar ex-

periences with "botnet" detectors, for example significant behavioral diversity (in their case arising from heterogeneous computing environments) and lack of ground truth. However, where they see synthetic threat data as a necessary evil resulting from a lack of real data, we see it as essential even if (and especially if) an abundance of real data is available. They correctly point out the risk of artifact arising from the intermixing real and synthetic test data; we can go further and say such artifacts are inevitable. We discuss the steps we take to avoid artifacts, but no doubt this is an area where much remains to be done. Rossow et al provide broad and at times quite specific recommendations on the design of malware experiments [7]. The guidelines they offer are quite useful but apply more to experiments in computational rather than social threat domains. For example, there is little doubt that *correctness* and *realism* of malware test data can be established in various ways; however, the same can not be said of insider threat test data, where reality itself (institutions, norms, ethics) is contingent and socially constructed [8].

Turning to the generation of insider threat test data, Glasser and Lindauer use an agent-based approach to generate both background and threat data [4]. They adopt a closed world approach to realism, defining reference measures for a range of data features; however, they also report difficulty in obtaining "realism" in more than a few dimensions at a time, and conclude that synthetic data can serve engineering, but not scientific purposes; a conclusion reached for the general case by Berk et al [3]. The Threat Stream Generator (TSG) [10], used to produce data for the VAST challenge, also uses narratives to describe test scenarios. As in our approach, generated threats are overlaid on background data that may itself be generated, or may be acquired from a real environment. And also in our work, special attention is paid to the roles played by characters in the threat narrative, and a premium is placed on social realism. That said, our work differs in several important ways. First and formemost, our methodology is designed to operate in an open loop. By design of the experimental protocol, the test team does not have access to the output of the analysis tools to determine whether generated data is realistic and defect-free.

3 Test Environment and Method

3.1 Preliminaries

Using terminology introduced by Sommer and Paxon the detectors for which we produce test data can be described as machine-learning based anomaly detectors rather than far simpler *violation* detectors [9]. The research objective is to detect *early* indicators and *precursors* of novel malicious user activity. For example, an unusual pattern of laptop computer crashes might indicate the early (experimental) stage of a more involved plan of sabotage, while a sequence of increasingly hostile email exchanges between employee and supervisor might indicate precursors to a decision by an employee to engage in sabotage. Producing test data that expresses early indicators and precursors requires threat models that are significantly more complex in their description of social behavior than those required for violation detectors. This raises the question: how do we define qualities of test data such as "plausibility" and "realism," especially where these qualities pertain to models of threat precursors such as disgruntlement, jealousy, rivalry, depression or other complex social phenomena? We offer no objective measures, but instead offer this observation: the strictures of *good* narrative fiction, and character development in particular, have proven in our work to be practical and effective in achieving plausibility. The use of fictional narrative may be theoretically justified as well: there is reason to believe that humans are "hard wired" to use fictional narrative to *abstract and simulate* social reality [6]—which is, after all, a fair description of the task at hand.

There is a risk that any threat model sufficiently complex to generate threat data rich in social behavior might also constitute a hidden source of sampling bias. To illustrate, Young *et. al.* (a research team that we supply with threat data), describe how counterintelligence expertise can be modeled and used to make better use of a portfolio of detectors, and to better interpret the results of these detectors [11]. They define a $(Stage \times Goal)$ model of threat scenarios with $Stage = \{exploration, experimentation, exploitation, execution, escape\ and\ evasion\}$, $Goal = \{destruction, misuse, corruption, theft\}$, and an expert interpretation for each *(Stage, Goal)* pair. While it is reasonable to incorporate such models into the detectors, problems arise when test data becomes biased by tacit sharing of such domain models by data producers and consumers. Which detector-specific domain models should we incorporate? What if research teams incorporate different domain models in their technologies? There is of course nothing problematic in incorporating domain-specific knowledge into detectors, nor with agreeing on such models in the design of experiments as a conscious decision to bound the scope of research. However, such scope restrictions were not desired in our case; in fact, the *mere awareness* of such models by data producers was regarded as a *grave threat to research validity*.

Lacking a better alternative, the research program imposed an unusual (and possibly unprecedented) test protocol, in which we are asked to provide test data without prior agreement on: (a) what constitutes threat precursors (e.g., disgruntlement?), or threat indicators (e.g., in-

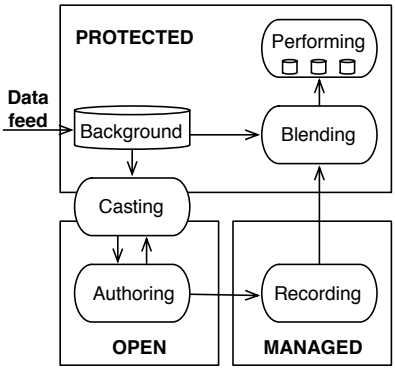

Figure 1: **Production Process and Environment**

creased use of printers?); (b) what features in data are observed by detectors, or how those features are mapped to precursors or indicators; (c) the anomaly scores obtained for any given threat; (d) how users are labeled (indeed, whether there are labels at all); or (e) the interpretation of scores as labels (i.e., "test results."). This "open loop" protocol does have the merit of avoiding overt sources of sample bias; but it is not testing in the conventional sense of the term.

3.2 Technical Details

Figure 1 depicts the technical environment and the major elements of our approach to test data production. Five main components of the production process are illustrated and described, below: authoring, recording, casting, blending and performing. The processes operate in three different environments: *protected* refers to the secure data facility, which imposes strict control on inbound and outbound data; *managed* refers to a recording studio, which for practical reasons (e.g., to ensure consistency of simulated and operational environments) is operated by the same industry partner that provides the data; *open* refers to any environment in which scenarios are authored.

3.2.1 Background Data

An industry partner provides background user data collected by host monitors installed on company-owned computers. The data is stripped of personally-identifying information before being streamed to a closed research facility where other safeguards are provided to protect user privacy. The data represents the activity of of approximately $5,000$ users, with approximately 10^7 monitoring events per month collected since the June 2012, with plans to continue until at least January 2015.

The data includes use of programs and desktop applications, including generic actions such as starting and stopping applications, as well as application-specific actions, such as sending and receiving email and internet browsing. Metadata is also provided, for example the content of email and instant messages, and the URLs of sites visited from a browser. The data includes other user actions, such as file copies. Host monitoring data is augmented with organizational metadata, such as business sites, locations, departments, and reporting hierarchy.

Background data establishes two kinds of ground truth. First, the data reflects real users engaged in real work and therefore establishes a *de facto* (if not fully comprehended) standard of realism. Second, all users are labeled as "benign." This second form of ground truth is imposed as an experimental control to protect the rights of users in the original monitored environment. It is difficult to assess the experimental impact of this decision; we assume (without much justification) it to be negligible under the assumption of a 0.02% base rate of malicious insiders. The data also constitutes an agreement between the producers and consumers of test data on what user actions are observable; and, in somewhat more detail, it defines a map from user-initiated actions to concrete features in background time-series data.

3.2.2 Threat Publication Protocol

Threat data are released in windows with one month of data. Each window contains between 3 and 5 test cases, and test cases must be completely observed in their release window. Test cases are independent; sequels and multi-window story arcs are not permitted. Users are sampled without replacement within windows; that is, a user may play a fictional role in only one test case per window. However, users are sampled with replacement over windows; that is, the same user may play roles in multiple test cases, provided they are in different windows. Background data can not be deleted, hidden, or modified in threat data; synthetic data can be inserted into background in "overlay" style only. Synthetic threat data is visible only in the current threat window but background data from previous threat windows is visible at all times; this means that researchers may choose to limit training data to the current threat window; on some or all previous threat windows; or on some combination of previous and current threat window.

3.2.3 Drama to Data Production Process

The data production process works as follows. Threats are scripted as fictional dramas with well-developed characters. Scripts are translated into programs that simulate fictional characters in a recording environment con-

figured to match the host-base sensors and collection policies of background data. Real users are selected to play the role of fictional characters, at which point their real background data is augmented with previously recorded synthetic character activity. A *central casting* service models features of background users (e.g., job roles and skills; friendship networks; work habits; computer preferences), so that users can be cast in fictional roles in a way that minimizes artifacts ("seams") between real and fictional user behavior. A single dramatic performance by a cast of modified users ("in character") and unmodified users ("extras") in a given threat window constitutes one test case; additional performances/test cases can be produced by varying the cast, or by the same cast performing in different threat windows.

The production process is described sequentially but is in fact is aggressively iterative: authoring is informed by the actors available for casting, and casting features evolve to meet the needs of new stories; details in the recording environment impose limitations on what can be presented, as well as opportunities for the equivalent of special effects; discoveries in the final production stages of blending have triggered rewrites. This is consistent with previous reported experience with narrative specified test data [10] and (we think not coincidentally) with many forms of theatrical production.

Authoring

Narratives describe sequences of causally-related events arising from the interaction of intentional agents; threat narratives are written in the style of fictional crime dramas, in the genre of insider threat. Insider threat dramas include conventional plot devices as well as genre-specific devices, such as methods of data exfiltration, deception and other forms of tradecraft; and genre-specific and character types, such as subject, co-conspirator, enabler, and victim. Each drama simulates a threat in an established genre, such as information technology sabotage, theft of intellectual property, fraud and espionage.

The design intent underlying all threat dramas is to make visible to the threat analyst, through directly observed user activity *or from plausible inference* from observed activity, whatever would be needed by the analyst to infer the *predicate*–a term of art in counterintelligence that refers to a hypothesis of malicious behavior, attributed to a subject, to be referred for further investigation or prosecution. We highlight "plausible inference" because it is a powerful tool for varying the level of difficulty in test cases we present; the lack of an observable might indicate evasion or sabotage, for example. In some cases we are required to sacrifice a small measure of plausibility to ensure that certain actions are observed, or that the expository clues of character intentions are revealed (e.g., the subject sending email to a co-conspirator with instructions to not use email to communicate). As described in AI literature, this is not a limitation of our method per se but of narratives that are described "top-down" rather than from the "bottom-up" way characters would interact in real life [2].

Dramas are written (mostly) in natural language, as a script having several distinct parts. A plot summary introduces the main characters and their motives, and highlights significant dramatic events, including the overt malicious act, and the predicate. Characters names follow the convention of criminal case files, such as "subject" and "co-conspirator." Each character may in addition have with one or more *casting constraints* or *casting preferences* (described in §3.2.3) that are used to select users from background to play the part of the corresponding fictional character. Dramatic action is described, more or less conventionally, as monologues (users interacting with programs to perform tasks) and dialogues (users interacting with each other using email or instant messaging) unfolding in a series of scenes.

While we have made use of insider threat case histories, there are, unsurprisingly, as many sources of inspiration for insider threat stories as there are for other kinds of story. Contemporary events have proven to be a reliable source. One good example of a threat story ripped directly from headlines[2] is *The Outsourcer's Apprentice*:

> A software developer outsources his job to China and spends his workdays web surfing.

Another and more interesting source of story inspiration is background data itself. In retrospect this is also not surprising: the workplace is not immune from the dramas that constitute the bulk of human social reality. Evidence of real-life dramas can be found in background, and some of these by their nature may inspire a threat story. One story that offers a compelling demonstration is *Layoff Logic Bomber*, which was inspired by email and IM chatter discovered in background data:

> An engineer worried about rumors of impending layoffs feels that he needs an "insurance policy" in case he gets laid-off. He creates a "logic bomb" which will delete all files from a number of company Linux systems in five days, unless he resets the timer before then.

The drama makes good use of real-world drama as a foundation for realistic synthetic threats. We were successful in finding a user to perform the role of logic bomber: he performed sysadmin-like activities (had the right skills), engaged in chatter about the layoff (his background activity coincides with the fictional plot) and exhibited no activity after the second week of the threat window (consistent with someone who has lost their job).

It is difficult to imagine a better example of exploiting real-life drama as an extensive fabula for fictional threat dramas.

Recording

Scripts are hand-translated into a "director" program that performs fictional character activity with simulated users in a recording studio; the translation step could be automated, but it is not a significant bottleneck since threat dramas are not overly long and the process of translating them into code is quite routine if tedious.

The recording studio is a remotely-accessible environment operated by the same partner that provides background data, and consists of email and directory servers set up to mimic the operational environment, and workstations hosted on virtual machines. The same host-based sensor that monitors the operational environment monitors each workstation, and care is taken to ensure that the same version of both the sensor and the monitoring policy are deployed in the recording and operational environments.

The director is operated by a red team member connected to the recording environment using a remote desktop sharing tool to access a master workstation. That workstation, in turn, connects to each of the monitored workstations using VNC. After executing the "director" and allowing the sensor to record the data, the contents of the recording environment's database are transferred through a de-identification process into a holding area in the program's research lab.

It is worth highlighting the following heuristic: place all code that simulates users in the same layer of abstraction as those users, and let the sensor generate the data. This has worked quite well: all but one data artifact reported turned out to be a systematic error in the collector. Our automation also operates at the level of the hypervisor; no control software needs to be installed on the monitored systems, and therefore control actions are invisible to the host-based sensors, thereby eliminating another significant source of artifact.

Casting

As is the case with dramas performed on stage on screen, casting decisions can make the difference between success and failure in producing dramas. As a concrete example, consider how we might select a user from background to play the role of the malicious user in *Layoff Logic Bomber*. The script calls for someone with systems administrator and programming skills. Choosing a user who never uses programming tools to play the role of bomber would likely result in unintended anomalies–social artifacts–arising from the sudden and inexplicable demonstration of hacking skills. Conversely, choosing a user who frequently engages in programming activities would, at least on this one dimension of user behavior, avoid that particular social artifact. This reasoning clearly generalizes; in fact, another criterion we used for selecting a user to play the role of logic bomber was that the user exhibit activity in the first two weeks of the window but none in the last two weeks, indicating (by inference) that the user had indeed been laid off.

Other examples of "dramatic features" include: a user is a software developer; a user is out of the office at certain times;' two users frequently chat with each other using an instant messenger program; and three users work at the same site. We distinguish two kinds of dramtic feature: "character features" describe individual users, for example their individual work hours, genders, job roles, etc.; "social features" describe relations among users such as: reporting hierarchy and friendship networks. Decision procedures can be defined for some features, for example internet browsing habits; other features can only be reliably obtained by manually inspecting data, for example whether a user has authority to sign business contracts. Both kinds of feature are used extensively in user selection.

Dramatic features are specified and stored (some automatically on a nightly basis, others inserted manually) in YAML format. Each feature is specified in two levels of abstraction. At a high level are dramatic features (both character and social) as illustrated above; each such dramatic feature has one or more concrete interpretations as "data features." For example, the character feature *removable-drive-user* has two distinct interpretations as data features, *two-stddev-above-mean* and *at-least-100-times*. Because we model social features as binary relations, we also specify for each feature its reflexivity, transitivity, and symmetry, and require these properties be preserved by all concrete interpretations.

Dramatic features are associated with characters in threat scripts; some features are regarded as hard constraints, others are soft constraints (preferences). Hard constraints are those features that a real user must satisfy to play a certain role; satisfying soft constraints are considered desirable but optional.

Blending

Blending is the transformation of recorded scenarios to make them appear as if the real user cast performed the scenario's fictional activities. Blending includes various transformations on data: inserting data recorded by enacting the drama on a separate testbed; changing the putative time of inserted events so that they appear to be sequential with existing user data; and, replacing details unique to the dramatic characters' recorded data with

corresponding details specific to the cast of user actors (for example, user id in data records, salutation and signature in user email). Blended data is combined with background data, and delivered to detectors as a simulated dramatic performance in threat windows.

The written script, its encoding in a director program, the captured synthetic data, dramatic feature (extraction logic and user labeling), and blending and verification code can be reused as is or modified to create variations of the original performance. Such variations may include a simple selection of a new cast, new time window of insertion, changes in dialogue or other details supporting the selected cast, changes in dialogue or other details requiring new casting criteria, or more extensive changes such as the number or even kinds of behaviors undertaken by the actors.

4 Evaluation of Results

Our stated objective is to produce threat data that is rich in social complexity, exhibits few artifacts, and is plausible and realistic. This objective, however, must be obtained without benefit of established theories of social realism or fully-established (or agreed) threat models, and our results are colored by this underlying reality. Most notably, without the benefit of traditional "closed loop" engineering feedback we lack any objective basis for several "claims" (sic).

In terms of traditional productivity measures, the techniques we describe were remarkably effective. Since July 2012 we have developed more than 60 performances of thirty 30 dramas in 20 threat windows, covering theft of intellectual property, industrial sabotage, and fraud and espionage; without schedule slip.

We lack satisfactory measures to substantiate claims we make about the realism or plausibility of test data we produce. Of course we can, and where useful we have, defined measures of correspondence between synthetic data and real data (in data features used for user selection, for example). However, these measures have meaning only within the production activity; they have little value without knowing how they relate to the detectors under test (which are black boxes to us) or to the results obtained by these detectors—which we are likewise discouraged from investigating for fear that we might "tune" data influence detection results. Heuristic support of realism and plausibility is quite strong, however:

- Realism: Each test case extends background fabula, and so leverages realism intrinsic in the data;
- Plausibility: Insider threat domain experts make substantial use of narrative; narratives they define are by definition plausible.[3]

Regarding how successful we have been in eliminating structural and social artifacts from test data, we again have only indirect (but still objective) measures: since August 2012 only one confirmed artifact has been reported; all others have turned out to have been a consequence of systematic error in the host-based collection system. While we do think there is a case to be made that narrative fiction provides a theoretical basis for thinking about realism and plausibility of test data, this is mostly speculative at this time.

We have only one indirect (though still objective) bottom-line measure of data quality: research teams have published more than seventy (and still counting) peer-reviewed articles describing technical results that directly depend on the test data we produced. While not entirely satisfying to engineers and scientists (including the authors), peer review is certainly a valid measure of value of threat data.

5 Discussion

Although we first adopted the drama metaphor merely because it provided convenient shorthand for writing stories and causing existing users to appear as characters in those stories, it has turned out to be important for solidifying and extending our understanding of our process. It led us, for example, to the decision to separate the roles of domain expert for providing a realistic basis for threat scenarios from script writer for providing realistic human interactions. The two-level specification of dramatic features may itself constitute a significant separation of concerns: it serves as an abstract interface to background data, allowing information about the data and its users to be safely exported from a protected data enclave, without risk to user privacy. This presents intriguing possibilities for sharing threat data that were previously unthinkable.

We also realized over time that the data from background users provided not just a canvas for our exogenous stories, but also a rich source for endogenous stories. The *Layoff Logic Bomb* scenario, discussed in §3.2.3, is an example of one such scenario where a story in the background data informed the construction of a top-level scenario. Endogenous stories allow us to create much richer stories exhibiting complex data characteristics that we could not hope to simulate, even if our vocabulary included deletion and modification of background records, not just addition. In the layoff case, one can imagine that rumors, stress, and disgruntlement, both before and after a layoff event, could have far-reaching effects to the observed characteristics of both individuals and the larger communication graph of the organization. By taking advantage of the native story, we needn't worry about failing to simulate those effects.

The decision to adopt an open loop test protocol is

well motivated, and judging from the results published by the various research teams, fully justified on practical grounds alone. One consequence of this is that researchers have no way of signaling to data producers their assessment of the quality of data. While the authors do not know of any better alternative to the open-loop approach adopted, we think there must be less radical ways of avoiding the particular threats to validity that motivated the approach. Ideally we would establish a formal feedback regime with the detection teams, with clear boundaries on what is and is not acceptable for discussion. In the meantime, in the absence of measures of quality or indicators of artifacts that might be misleading unsupervised anomaly detectors, the best we can do is continually refine our entire process, paying specific attention to understanding the context around observed phenomena in data to improve blending.

6 Conclusion

It is difficult to see how we could have adopted conventional testing objectives under open-loop test conditions, or the common sense objective of "estimat[ing] the 'real world' performance" of detectors put forth by Killourhy and Maxion [5]. Neither would it have made sense for us to act as adversary of research teams in a penetration-testing style zero-sum game. Simply stated, under these conditions, the red team always wins. We can find ways of making user actions unobservable for the given (or any) collection policy; and can produce threat data that is valid in the domain construct and narrative form but with no observable events. This, however, would constitute a test of the collection policy, not the threat detector. In a few scenarios we do bypass the collection policy to simulate malicious actors who possess espionage tradecraft; in this case what we wish to be observed is the possession tradecraft, and it is the fictional character, not the red team, that is acting as adversary.

Gradually, but ultimately, we came to appreciate the position adopted by Sommer and Paxon [9]: "When evaluating an anomaly detection system, the primary objective should be to develop insight into the system's capabilities." We think that embracing fictional narrative as a fundamental technique for abstracting and simulating the complexities of social reality advances this objective.

7 Acknowledgments

Thanks to Marla Landers-Renouf for dramatic scripts; and David Stein, for helping us understand the collection system. This material is based upon work funded and supported by the ADAMS (Anomaly Detection at Multiple Scales) Program of the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center sponsored by the Department of Defense. The findings and conclusions expressed are those of the authors and do not necessarily represent the views of the U.S. Government. This material has been approved for public release and unlimited distribution.

References

[1] AVIV, A. J., AND HAEBERLEN, A. Challenges in experimenting with botnet detection systems. In *Proceedings of the 4th Conference on Cyber Security Experimentation and Test* (Berkeley, CA, USA, 2011), CSET'11, USENIX Association.

[2] AYLETT, R., LOUCHART, S., AND WEALLANS, A. Research in Interactive Drama Environments, Role-Play and Story-telling. In *LNCS* (2011), vol. 7067, Springer Verlag, pp. 1–12.

[3] BERK, V. H., GREGORIO-DE SOUZA, I., AND MURPHY, J. P. Generating realistic environments for cyber operations development, testing, and training. In *SPIE Defense, Security, and Sensing* (2012), International Society for Optics and Photonics, pp. 835908–835908.

[4] GLASSER, J., AND LINDAUER, B. Bridging the gap: A pragmatic approach to generating insider threat data. In *Security and Privacy Workshops (SPW), 2013 IEEE* (2013), IEEE, pp. 98–104.

[5] KILLOURHY, K., AND MAXION, R. Toward realistic and artifact-free insider-threat data. In *Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual* (Dec 2007), pp. 87–96.

[6] MAR, R. A., AND OATLEY, K. The function of fiction is the abstraction and simulation of social experience. *Perspectives on psychological science 3*, 3 (2008), 173–192.

[7] ROSSOW, C., DIETRICH, C. J., GRIER, C., KREIBICH, C., PAXSON, V., POHLMANN, N., BOS, H., AND STEEN, M. V. Prudent practices for designing malware experiments: Status quo and outlook. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy* (Washington, DC, USA, 2012), SP '12, IEEE Computer Society, pp. 65–79.

[8] SEARLE, J. R. *The construction of social reality*. Simon and Schuster, 1995.

[9] SOMMER, R., AND PAXSON, V. Outside the closed world: On using machine learning for network intrusion detection. In *Security and Privacy (SP), 2010 IEEE Symposium on* (May 2010), pp. 305–316.

[10] WHITING, M. A., HAACK, J., AND VARLEY, C. Creating realistic, scenario-based synthetic data for test and evaluation of information analytics software. In *Proceedings of the 2008 Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization* (2008), ACM, p. 8.

[11] YOUNG, W. T., GOLDBERG, H. G., MEMORY, A., SARTAIN, J. F., AND TED, E. Use of domain knowledge to detect insider threats in computer activities. In *Security and Privacy Workshops (SPW), 2013 IEEE* (2013), IEEE, pp. 60–67.

Notes

[1] The authors have heard counterintelligence experts use this as an *upper bound*; however, we know of no reliable data on this.

[2] See `http://www.bbc.co.uk/news/technology-21043693`

[3] Despite appearance, not circular.