Bridging the Data Gap: Data Related Challenges in Evaluating Large Scale Collaborative Security Systems *

John Sonchack[α], Adam J. Aviv[β], and Jonathan M. Smith[α]
[α] *University of Pennsylvania* [β] *Swarthmore College*
{jsonch,jms}@cis.upenn.edu aviv@cs.swarthmore.edu

Abstract

Data-sharing approaches such as *collaborative security* have been successfully applied to systems addressing multiple classes of cyber security threats. In spite of these results, scale presents a major challenge to further advances: collaborative security systems are designed to operate at a large scale (Internet- or ISP-scale), and obtaining and sharing traces suitable for experimentation is difficult. We illustrate these challenges via an analysis of recently proposed collaborative systems. We argue for the development of simulation techniques designed specifically to address these challenges and sketch one such technique, *parameterized trace scaling*, which expands small traces to generate realistic large scale traces sufficient for analyzing collaborative security systems.

1 Introduction

The technique of *collaborative security*, where information is shared to enhance defenses, has been applied to several classes of cyber security systems such as anomaly detectors [11], blacklist generators [31], and worm signature generators [16], as well as more generally in other domains [19, 30]. The promise of such systems is great: with a broader view of network threats, more targeted defenses can be utilized. Further, networks similar to those previously attacked can preemptively install appropriate IDS or firewall rules for proactive defense, limiting and isolating incursions before they are observed locally.

The results from such systems show the high potential of collaborative security, and further advancements should be sought. Unfortunately, significant challenges exist to broad access of adequate experimental resources for evaluating collaborative security systems. Collaborative systems are designed for large-scale deployment, either at Internet or large-ISP scale. Effectively evaluating a system of this size requires access to large scale network traces, or data equivalent to such traces. Further, these traces must also contain active threats so that the security components of the system can be exercised.

*This work was supported by ONR Grant N00014-12-1-0757.

Obtaining (or collecting) suitable traces can be extremely difficult. Once obtained, disseminating them broadly across the research community is nearly impossible, due to the sensitivity of real network traces. The difficulty of obtaining suitable traces effectively locks out many interested researchers from evaluating new collaborative systems, while the limitations on sharing underlying experimental data hinders the scientific process and slows the advancement of collaborative security techniques. It is important to bridge the *data gap* to working with large scale network traces.

We begin by illustrating the challenges to research progress in collaborative security by laying out properties of an "ideal world" for the experimental and scientific process. We use this as a baseline against which we analyze three case studies of the data challenges faced in the evaluations of proposed collaborative security systems [31, 16, 28]. We *do not* argue that the proposed systems or their evaluations are flawed; on the contrary, the results are encouraging and positive. Rather, the goal of this analysis is to demonstrate the barriers to evaluation and testing, and thus to research progress.

Based on this analysis, we argue that designing simulators specifically for evaluating collaborative security systems would help solve the *data gap*. Current simulation tools are not well-suited for this domain, such as virtual testbeds [23, 17] or traffic generators [29] because they do not provide direct control over the factors that are important to collaborative security systems, and have complex configurations that are difficult to validate against reality. The contribution we make, in addition to analyzing the underlying data issues and arguing for a simulation based solution, is the proposal of a new simulation technique that would take a first step towards overcoming the data gap. The *parameterized trace scaling* technique uses small traces as the basis for generating realistic, large scale simulations that can be highly parameterized to match different network settings. Such a simulator can offer many benefits, and could be used to validate and extend previous results as well as to evaluate new systems.

2 Experimental Ideals

In this section, we discuss four scientific ideals that we believe are important for experimental procedures in the collaborative security domain: (1) experimental reproducibility, (2) experimental control, (3) ground truth, and (4) experimentation at scale. As ideals, these goals are difficult to achieve, particularly in the context of large-scale security experimentation. The goal of this section is to provide a discussion of both the ideals and the challenges associated with them.

2.1 Reproducibility

A key component of the scientific process is *reproducibility*, which enables future research to build off of previous results by reproducing them using the same procedures and input parameters. Reproducibility is particularly challenging to achieve in collaborative security research because of the sensitivity of underlying experimental data, especially network traces. First, it is difficult to acquire network traces because network operators are reluctant to share network traces for experimentation, as they may contain customer information. Second, even if data is acquired, it is difficult to receive permission to redistribute the traces.

Privacy concerns are the major reason for operator reluctance because network traces may contain records of private communications. The proper scrubbing and anonymizing of network traces is a delicate and challenging procedure [20] that does not always sufficiently protect network and end-user privacy. It has been shown that releasing anonymized data still presents risks [12] as well as possible legal challenges [27].

There are two viable alternatives for researchers pursuing network data that allow for easy experimental reproduction. First, researchers could perform experiments using data from publicly accessible repositories such as CAIDA [1], DShield [3], or PREDICT [7]; however, such repositories generally contain only anonymized and scrubbed data, which limits certain evaluations, such as those of techniques that perform payload inspection. Second, researchers could use virtual testbeds such as NS [23] or Mininet [17]; however, traffic generated on such systems may not be realistic because they require exact, user-defined network connectivity information on a per device level that is difficult to validate.

2.2 Experimental Control

Collaborative security systems operate in complex environments where there are many factors that can affect performance. The heterogeneity of a typical network environment necessitate experiments that consider the effects of varied factors on the results, which requires careful *experimental control*, or the ability for an experimenter to minimize the effects of factors aside from those being measured. Experimenting with traces gathered from real networks limits experimental control in two ways. First, a trace only provides a sample of the network environment from a single vantage point at a single moment, which limits the ability of researchers to generalize their results, particularly in large scale settings. Second, security factors, such as whether a host is infected when it generates certain packets, are difficult to identify using only traces, and thus even harder to control.

A common technique for increasing experimental control is *trace synthesis* [13, 14, 19], in which a background trace is mixed with a trace of malicious traffic, such as from a honeypot [9, 22]. This technique fails because synthesized traces may contain unintended artifacts. For example, network traces of a worm or botnet collected in a testbed may not be the same as a trace collected in a live network infection. These artifacts would render the malicious traffic easier to detect in synthesized traces.

An alternative to trace synthesis is *network simulation*; however, existing simulation tools are not well-suited for evaluating collaborative security systems. Virtual testbeds [23, 17] allow users to control low level factors such as network link properties and the code executed on devices, but not higher level factors which are important to collaborative security, such as inter-network correlation. Traffic generating tools [29] provide control over statistical properties of packet, such as packet inter-arrivals, but do not generate packet payloads, which are necessary to experiment with many collaborative cyber security systems.

2.3 Ground Truth

The goal of many collaborative security systems is to identify malicious traffic, and the evaluation metrics used, such as *detection rate* (the percentage of threats identified) and *false positive rate* (the percentage of non threats identified as threats), require detailed knowledge of what is and what is not malicious traffic. Measuring accuracy with such metrics requires *ground truth*: knowledge of which evaluation traffic is malicious.

Determining ground truth for network traffic is difficult even with access to the end host. In most cases researchers do not have direct access to the end hosts in a trace because the network domains are outside the researchers' control. Further, many publicly available traces are not current, and network configurations may have changed, even if end host access was reasonable. Most publicly available traces are also incomplete for privacy reasons and have payloads removed, source addresses anonymized, or are sampled, which makes determining ground truth even more of a challenge.

Due to the challenges of determining ground truth, many evaluations use metrics that are not based upon it, focusing instead on measurements such as response time or the volume of traffic detected [31]. These metrics are useful but cannot measure detection accuracy. Alternately, some evaluations use an existing detection technique to determine ground truth [16] and then measure performance of the new system based on a comparison. Unfortunately, the accuracy of such partial ground truth depends on the secondary detection system, which may have both false positives and true negatives.

2.4 Experimentation at Scale

Collaborative security systems are designed to leverage information gathered from multiple networks. The scale, or the total number and size of the participating networks, can greatly affect the results, and important questions about collaborative security systems can only be answered by *experimentation at scale*. That is, the experimentation should use a data set that is large enough to reflect the scale for which the system or technique was designed.

Experimenting at scale compounds the challenges of the previously discussed ideals. Reproducibility is now particularly difficult to achieve because multiple networks would need to release data. Experimental control and ground truth continue to be problematic because network data at this scale is harder to analyze thoroughly. There are publicly available data sets and simulation techniques that are applicable for large scale experimentation; however, as previously described, there are serious drawbacks.

3 Case Studies

In this section, we analyze the practical challenges of collaborative security experimentation. We present three case studies that examine proposed collaborative security systems: Highly Predictive Blacklisting [31], Autograph [16], and an Internet scale detection technique [28]. These systems are promising, and their evaluations show the potential for collaboration as a network security technique. However, there are common data related challenges to meeting experimental ideals in the evaluations of these systems. We examine how well each evaluation meets the experimental ideals and identify the underlying data related challenges. We summarize the results of our analysis in Table 1.

3.1 Highly Predictive Blacklisting

Collaborative blacklisting systems generate IP address blacklists by analyzing logs submitted by participating networks. Highly Predictive Blacklisting [31] computes a *relevance* and *severity* score for each IP address that appears in submitted logs, and then combines these scores to generate a blacklist specific to each collaborator. This approach to blacklist generation is based on the observation that attackers tend to target small, stable sets of networks [15]. Therefore, generating effective blacklists depends not only on determining which IP addresses are malicious, but also who they are likely to attack.

The evaluation of this system used a set of over 700 million log entries from intrusion detection system (IDS) submitted to DShield [3] by more than 1500 networks over a two month period of time. System performance was measured primarily on *hit count*: the number of IP addresses blocked by the blacklist. With respect to this metric, the evaluation showed that the Highly Predictive Blacklisting system produced much better blacklists than other common heuristics for blacklist generation.

Reproducibility: The data used to evaluate the Highly Predictive Blacklisting system was provided by DShield [3], a community-based IDS log correlation system. DShield allows researchers to obtain data for non-commercial purposes, provided they agree to certain standards, such as protecting the data from improper disclosure [5]. Since most researchers can obtain data similar to that used in the original evaluation, the results are likely reproducible.

Experimental Control: The evaluation controlled for two factors: time and network perspective. The dynamics of the Internet dictate that the alerts submitted to the blacklist system will vary over time. To control for temporal changes, the evaluation splits the 60 days of experimental data set into 12 intervals and shows that system performance is consistent across the intervals.

Internet traffic is also heterogeneous across networks. A large academic network, for example, is likely to carry different traffic and generate different IDS alerts than a corporate network. The performance of blacklists generated should depend on the participating network. The evaluation controls this factor by analyzing the distribution of blacklist performance across contributing networks.

However, an important factor that is not controlled in this evaluation is attacker behavior. Highly Predictive Blacklisting is motivated by the observation that attackers target small, stable groups of networks [15]. Without control for attacker behavior, it is difficult to determine whether the system accurately models this phenomenon. Controlling for this factor requires an understanding of the attacker behaviors present in the underlying data. It is difficult enough to determine which records correspond to attacks (see below), let alone accurately track specific behaviors.

Ground Truth: Unfortunately, there is no ground truth in the data used in experimentation because it cannot be

determined directly if an IP on a blacklist is truly malicious. Ground truth is required to measure blacklist accuracy with metrics like *detection rate* and *false positive rate*. Without ground truth, these standard accuracy metrics cannot be measured, and instead, the evaluation focuses on *hit count*, a metric that does not reflect accuracy.

The lack of ground truth in this evaluation can be entirely attributed to the data set used, which consists of over 700 million IDS logs. An IDS alert does not necessarily imply a malicious event as false alerts are common, and it is impossible to investigate an alert further without, at the very least, access to the traffic that triggered it.

Evaluation at Scale: All experiments in this evaluation are done using a very large scale data set containing logs from over 1500 networks. The authors of this system even went beyond the ideal of evaluation at scale, by implementing and deploying a prototype system that is still accessible today [4].

3.2 Autograph

Autograph [16] generates signatures that an IDS can use to detect worm traffic. The system operates in two phases: a *suspicious flow collection phase* that gathers network flows from hosts suspected of propagating worms, and a *signature generation phase* that automatically derives IDS signatures from the collected flow payloads. Autograph deployments collaborate by sharing the IP addresses of hosts suspected to be infected, which enables faster responses to outbreaks.

Autograph is evaluated in two parts. The first part analyzes the quality of signatures generated by Autograph using network traces collected from large lab networks. The second part of the evaluation analyzes the benefit of collaboration in Autograph using a simulation based on models found in [18].

Reproducibility: The evaluation of Autograph's automatic IDS signature generation is difficult to reproduce. The network traces used to evaluate the signature generator were collected on a large laboratory network with full packet payloads. The traces used in these experiments were not made publicly available, and due to the dynamic nature of the Internet, it is unclear that even a similarly acquired network trace would have the same characteristics as the original analysis. As noted previously, just collecting a trace of this scale, regardless of network characteristics, is challenging in its own right. Unfortunately, evaluations of systems like Autograph that require inspection of packet payloads are extremely difficult to reproduce. The original data is too sensitive for public release; already publicly available data sets are insufficient due to a lack of payloads; and, simulated traffic generators do not produce reasonable payloads.

The collaborative side of Autograph is evaluated using a worm propagation model, which is reproducible with well-specified parameters. The simulation only models the probing characteristics of a worm outbreak and not the packet level characteristics. Thus, the performance of the system, in total, depends on both the ability to generate IDS rules and properly distribute them, which the reproducible simulation alone cannot provide.

Experimental Control: The IDS rule generator will be directly affected by worm behavior; however, only one aspect of worm behavior was controlled, the amount of randomization in worm exploit code. Other worm propagation behavior, such as probing targets, infection vectors and infection rate, were not part of the experimental control, which is understandable given the challenge of collecting live traffic with security threats. Even still, the evaluation presents strong evidence that Autograph can generate effective signatures for several worm varieties, but without proper controls it is difficult to determine if there are kinds of worm behavior that Autograph performs well (or poorly) against.

The second part of the evaluation, which analyzed the benefit of collaboration for decreasing response time to a worm outbreak, had more experimental controls. These experiments were done using a model based simulation, and so all of the factors that affected system performance were directly controlled as input to the model.

Ground Truth: Autograph used a *partial ground truth* to evaluate the detection rate of the system. The partial ground truth was provided by replaying the collected traces through the Bro [21] IDS; that is, if Bro with its signatures declared a network flow as malicious, it was considered as part of the ground truth in the evaluation of Autograph. Unfortunately, this method of determining ground truth is insufficient as the false positive rate of an IDS is already relatively high, and the authors of Autograph even mentioned several false alerts.

The primary limitation of partial ground truth is that it changes the meaning of accuracy metrics. Instead of being absolute measures of performance, they become measures of performance relative to a reference system. For example, detection rate measured using partial ground truth does not illustrate how many threats a system can detect, but rather how many threats a system can detect *out of the threats detectable by the reference system.* However, for evaluations such as this, which analyze a system's performance across millions of flows previously collected at multiple networks, the choice is not between partial ground truth and absolute ground truth, but rather between partial ground truth and no ground truth. Between these two choices, it is understandable to choose a

partial ground truth, which provides some accuracy metrics rather than none.

Evaluation at Scale: The collaborative aspect of Autograph is designed to share the IP addresses of hosts that independent deployments believe are infected. This is evaluated using a simulation that models the outbreak of a worm across $338,652$ vulnerable hosts. Unfortunately, this simulation is significantly less detailed than the traces used to evaluate the quality of generated signatures. Whereas those traces contained both malicious and innocuous traffic, this simulation only generates records for malicious events relating to worm scanning and propagation. It does not model any innocuous communications, which can affect the response time of the system, nor does it contain payloads, which means that the IDS rule generation aspect of Autograph is not evaluated at scale.

Although the simulation used in the evaluation limits experimental results, there are currently no other reasonable options for evaluating a system like Autograph at scale. It is almost impossible to collect traces from a large number of networks, publicly available traces do not contain the packet payloads needed for experimentation with signature quality, and there are no models for simulating innocuous events.

3.3 Internet Scale Detection

The effects of worm outbreaks, DDoS attacks, and other large scale network events can be felt across the Internet. Wagner et al. proposed a technique [28] for detecting such events by analyzing Internet scale traffic patterns. Based on the observations that large scale Internet events skew traffic features, the proposed technique attempts to correlate the actions of thousands of hosts distributed across many networks, and would be ideal for a collaborative extension. Wagner et al. use entropy to quantify the *skewness* of an event; for example, the outbreak of the Blaster worm [10] greatly increased the percentage of Internet traffic involving port 135, which in turn decreased the entropy of the frequency distribution of destination ports.

Wagner et al. evaluate their technique by detecting the outbreaks of the Blaster [10] and Witty [26] worms. They experimented on flow traces collected during the actual outbreak of these worms. All traces were collected from the SWITCH (Swiss Academic and Research Network) network, a medium-sized Swiss Internet backbone, which carries around 5% of all Swiss Internet traffic.

Reproducibility: Reproducing this evaluation is clearly difficult because it requires the actual traces used for experimentation, or new traces collected at this scale that coincide with a major worm outbreak. There are a few public data sets of similar scale and detail available to researchers in the PREDICT [7] and CAIDA [1] repositories. However, we currently know of no large scale publicly available data sets from an ISP size network that contain a large scale worm outbreak. The lack of reproducibility is particularly problematic for this evaluation because Wagner et al. are advocating an approach to anomaly detection rather than a system, which presents further challenges to independent researchers advancing the proposed technique in prototype systems.

Releasing the underlying traces is likely infeasible and outside the control of the researchers. The traces were from a large ISP, and contained over 60 million flows per hour. Thus, there are significant privacy and security concerns about releasing such data, for both the ISP and its customers.

Experimental Control: In order to control for worm behavior, the evaluation is broken into two sets of experiments performed on traces collected during the outbreak of different worms, the Blaster and Witty worm. The Blaster worm attacked vulnerabilities on TCP port 135 of hosts running Microsoft operating systems and infected about 200,000-500,000 hosts. The Witty worm, on the other hand, attacked firewall software over UDP, choosing target ports randomly, and only infected about 15,000 hosts. By including one set of experiments for each outbreak, the evaluation controlled for the effects of these different characteristics on the performance of the anomaly detection technique.

However, there are other important factors that were not controlled. The traces both came from the same ISP, so there was no control for vantage point, limiting the ability to analyze whether this approach would be applicable for anomaly detection at other, perhaps smaller, networks. Temporal changes in traffic are also not controlled: each set of experiments was only performed once, over the 24 hour period of time surrounding the worm outbreaks. While these controls would be ideal, it would be unreasonable to ask researchers to control for these factors because collecting just a single large scale trace with a worm outbreak is very challenging, collecting traces from multiple vantage points across federated network during a worm outbreak is even more challenging.

Ground Truth: The evaluation traces were collected during the outbreak of two quickly propagating worms that affected much of the Internet. These worms were already widely studied and easily identifiable. Thus, the ground truth of the traces is known.

Evaluation at Scale: The evaluation traces were collected at an ISP and contain a large volume of traffic from many different sources and destinations. Traffic observed at the interior of the network can differ significantly from the traffic generated at its edge due to network effects, such as routing asymmetry [6]. Although the traces are

Table 1: How well each evaluation meets experimental ideals, and how well an evaluation could meet experimental ideals using parameterized trace scaling.

Original Evaluations	Reproducibility	Experimental Control	Ground Truth	Evaluation at Scale
Highly Predictive Blacklisting	●	◐	○	●
Autograph	◐	◐	◐	◐
Internet Scale Detection	○	◐	●	◐
Parameterized Trace Scaling	●	●	●	◐

Legend: ○ : evaluation does not meet ideal. ◐ : evaluation partially meets ideal. ● : evaluation meets ideal.

large scale with respect to volume, they may not accurately represent the many distinct vantage points of the Internet. This would be required to extend the technique with collaborative elements that account for varied network perspectives.

3.4 Other Systems

The systems studied in this section demonstrate the benefits of collaboration in large scale security systems. Testing and advancing such collaborative techniques should be feasible for interested researchers, especially since these proposals are several years old. Some collaborative systems have been proposed more recently, such as Botgrep [19], which analyzes inter-host communication patterns to find botnets, and a web server anomaly detector [11] that correlates anomalies from different domains to reduce false positive rates. Unfortunately, collaborative systems remain uncommon and the evaluations of these newer systems face similar data related challenges and limitation, illustrating that experimentation with collaborative techniques has not become more accessible or ideal since the systems studied here were proposed.

Data challenges in large scale security experimentation are not limited to the collaborative domain. Others have detailed these difficulties in evaluating several classes of network security systems: Aviv et al. [8] discussed the challenges of experimenting with botnet detection systems, Rossow et al. [25] outlined prudent practices for designing experiments based on malware execution, and Ringberg et al. [24] argued that experimental techniques based only on network traces are insufficient for evaluating anomaly detection systems.

4 Collaborative Security Simulator

Unfortunately, many of the ideals outlined in Section 2 are not feasible given the current realities of computer security research. The public release of network traces will always be a privacy concern, and when such data is released, it is often insufficient for the task of evaluating large scale security systems. A ready solution to this issue is beyond the scope of this paper. The challenge of working with simulators, however, is more tractable.

The challenge lies in the fact that existing simulation tools are not well-suited to this style of security research, and, instead, have been designed for network protocol, routing, and congestion control experimentation. A different simulation approach for security related problems should be taken, where the simulator is specifically designed for experimentation with collaborative security.

In this section, we propose a simulation technique, *parameterized trace scaling* that takes a first step towards the goal of designing simulators specifically for collaborative security system research. Our technique extracts flow payloads from a small scale input trace, and then replays the flows between simulated hosts in an event driven simulation, where the events are generated using statistical models of network factors relevant to collaborative security systems. In this section, we describe the motivation for our technique and discuss its the potential benefits, limitations, and applications for collaborative security research.

4.1 Trace Scaling

Security techniques that rely on content filters often require examining full packet payloads, e.g., via deep packet inspection, and it is known that these techniques can facilitate faster and more precise defenses [18]. Thus, it is important that a cyber-security simulator is capable of generating traffic that contains full payloads to match the needs of these techniques.

One approach for generating payloads is simulating the individual network applications and protocols, as done by NS [23]. Unfortunately, this approach is problematic because even simulating a small network would require modeling a realistic set of application level protocols. Furthermore, due to the diversity and complexity of network applications, simulated applications may not generate reasonable payloads, which would introduce artifacts that could skew results.

Instead of generating payloads using functions that simulate network connected applications, we propose to re-use the payloads from an input trace. Flows from a small scale input trace could be classified based on the application that generated them; then, when a simulated host is configured to run a certain application, it could replay flows from that application's class. Such a *trace scaling* technique would be easier to configure, since users would no longer have to provide code to generate traffic

Table 2: Network factors can be modeled by calculating event attribute values using statistical distributions.

Factor	Statistical Interpretation	Event Attribute
Threat Prolificness	Frequency of hosts occurring as the source of malicious events	Source
Traffic Diversity	Frequency of payloads from each application occurring in events	Payload
Inter-network Correlation	Frequency of hosts occurring as the source of events at each network	Source

for each application. It would also provide the guarantee that payloads are realistic because they have actually been observed in real traffic.

4.2 Parameterization

Other factors besides payloads affect collaborative security systems, such as inter-network correlations, traffic diversity, and threat characteristics. Existing event driven simulators allow users to configure network devices, but do not allow them to tune simulations with respect to such factors. A simulator designed to analyze collaborative security systems should provide more direct control over factors important to collaborative security systems. We propose to achieve this by computing the attribute values of simulation events based on statistical models of network and Internet factors.

Table 2 provides examples of factors important to collaborative security systems, interpretations of the factors that allow them to be modeled with statistical distributions, and event attributes that could be calculated using these distributions. Users would configure a parameterized trace scaling simulator by providing the distributions or distribution parameters that modeled desired factors. In turn, the simulator would use the distributions to generate events and produce simulations with the desired properties. Configuration of such a simulator would be simpler than that of existing simulators, and allow for a direct investigation of how collaborative system performance varies with gradual changes to statistically modeled network factors. Furthermore, configurations would be verifiable against reality: distributions and parameters can be measured from live networks as well as tested and validated independently. Validation could even be done using traces that are not directly useful in collaborative security experimentation because of, for example, anonymization or payload removal.

4.3 Benefits and Limitations

Parameterized trace scaling would take a first step towards addressing the data related challenges in collaborative security system experimentation. It would facilitate reproducibility by enabling researchers to scale up small traces in a way that simulates much larger (and harder to obtain) traces underlying previous systems' evaluations. A more accurate ground truth would also be achievable because researchers could expand locally collected traces from networks under their control, and thus have access to the host level details required for determining ground truth. Experimental control would also be more straightforward due to simple parameterization. However, evaluations at scale would continue to depend on the network diversity of the input traces, and smaller network traces often contain less traffic diversity, which cannot be accounted for in the scaling technique. We summarize the benefits and limitations of parameterized traces scaling in Table 1, below we demonstrate how limitations in the previous evaluations outlined in Section 3 could benefit from this simulation technique.

Highly Predictive Blacklisting: The biggest data related challenge with the Highly Predictive Blacklisting system is a lack of ground truth, which limited the authors from reporting an accuracy metric. With the proposed simulator, this analysis could be extended using publicly available data sets where a ground truth is known. For example, the DARPA IDS evaluation data set [2] could be scaled up to simulate input for the blacklisting system to not only confirm previous results but also allow the researchers to report an accuracy metric since ground truth is known in the DARPA IDS trace.

Autograph: The data related challenges of Autograph are all due to the use of large scale traces, particularly the use of traces that require payloads. This limits reproducibility, experimental control, and ground truth. While some of these challenges were partially overcome by using a reference system to find partial ground truth and a propagation model to simulate a controlled worm outbreak, the use of parameterized trace scaling could fill these gaps further. The proposed technique will properly simulate traffic with both malicious and innocuous payloads, overcoming the large data trace challenges generally, and allowing the evaluation to broaden its experimentation to include an analysis of IDS rule generation with the worm outbreak modeling. However, issues of evaluating at scale will remain challenging because the trace scaling technique is still dependent on the traffic diversity of the input trace.

Internet Scale Detection: The primary data challenge of the Internet Scale Detection technique is the difficulty of collecting traces that coincide with Internet scale events. This limits reproducibility and hinders experiments with respect to issues of scale, such as controlling for large malicious events and evaluating the system's effectiveness at diverse network perspectives. Parameter-

ized trace scaling would enable researchers to design reproducible experiments with Internet-scale events, which would in turn enable new prototyping of the detection technique, but, again, the proposed trace scaling technique is limited in providing a broad network perspective without already diverse input traces.

5 Conclusion and Future Work

Collaborative security systems have broad applicability and great potential for addressing large scale network security threats, as demonstrated by several recently proposed systems [31, 16, 28]. Unfortunately, scientific progress in this area is stymied by a *data gap* where the requisite data to extend previously proposed systems and evaluate new systems is inaccessible to the broader research community. Acquiring, analyzing, and disseminating scale-appropriate traces is often difficult or impossible. This locks out many interested researchers and obstructs ideal experimentation. To address these issues, we argue that new simulators are needed that are specifically designed for the evaluation of security systems of this scale. We believe that *parameterized trace scaling* simulators would offer a solution through the process of scaling up smaller, obtainable network traces to simulate large scale traces, while also providing strong parameterized control over important environmental factors.

To this end, we have developed a small prototype of a parameterized trace scaling simulator, and, in preliminary experimentation, we have observed some of this technique's potential as a tool for collaborative security experimentation. We were able to reproduce and extend results from several of the systems analyzed in this paper using a scaled up version of the publicly available DARPA IDS evaluation trace [2]. These results suggest that parameterized trace scaling is a viable option for furthering cyber-security evaluation, allowing researchers to validate results, compare systems, and build on existing analysis without the need of collecting large scale traces.

References

[1] CAIDA Data Overview. `http://www.caida.org/data/overview/`.

[2] Darpa ids evaluation data set. `http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/`.

[3] Dshield.org. `http://www.dshield.org/`.

[4] Highly predictive blacklist. `http://www.dshield.org/hpbinfo.html`.

[5] ISC Research Feed. `https://isc.sans.edu/researchfeed.html`.

[6] Observing routing asymmetry in internet traffic. `http://www.caida.org/research/traffic-analysis/asymmetry/`.

[7] Protected repository for the defense of infrastructure against cyber threats. `http://predict.org`.

[8] Adam J Aviv and Andreas Haeberlen. Challenges in experimenting with botnet detection systems. In *USENIX 4th CSET Workshop*, 2011.

[9] Paul Baecher, Markus Koetter, Thorsten Holz, Maximillian Dornseif, and Felix Freiling. The nepenthes platform: An efficient approach to collect malware. In *RAID'06*, pages 165–184. Springer, 2006.

[10] Michael Bailey, Evan Cooke, Farnam Jahanian, David Watson, and Jose Nazario. The blaster worm: Then and now. *Security & Privacy, IEEE*, 3(4):26–31, 2005.

[11] Nathaniel Boggs, Sharath Hiremagalore, Angelos Stavrou, and Salvatore J Stolfo. Cross-domain collaborative anomaly detection: so far yet so close. In *RAID'11*, pages 142–160. Springer, 2011.

[12] Scott Coull, Charles Wright, Fabian Monrose, Michael Collins, Michael K Reiter, et al. Playing devil's advocate: Inferring sensitive information from anonymized network traces. In *NDSS'07*, pages 35–47, 2007.

[13] Jan Goebel and Thorsten Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 8–8. Cambridge, MA, 2007.

[14] Guofei Gu, Roberto Perdisci, Junjie Zhang, Wenke Lee, et al. Botminer: clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *Proceedings of the 17th conference on Security symposium*, pages 139–154, 2008.

[15] Sachin Katti, Balachander Krishnamurthy, and Dina Katabi. Collaborating against common enemies. In *ACM IMC*, 2005.

[16] Hyang-Ah Kim and Brad Karp. Autograph: Toward automated, distributed worm signature detection. In *Proceedings of the 13th USENIX Security Symposium*, pages 271–286, 2004.

[17] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.

[18] David Moore, Colleen Shannon, Geoffrey M Voelker, and Stefan Savage. Internet quarantine: Requirements for containing self-propagating code. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1901–1910. IEEE, 2003.

[19] Shishir Nagaraja, Prateek Mittal, Chi-Yao Hong, Matthew Caesar, and Nikita Borisov. Botgrep: finding p2p bots with structured graph analysis. In *Proceedings of the 19th USENIX conference on Security*, pages 7–7. USENIX Association, 2010.

[20] Ruoming Pang, Mark Allman, Vern Paxson, and Jason Lee. The devil and packet trace anonymization. *ACM SIGCOMM Computer Communication Review*, 36(1):29–38, 2006.

[21] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23):2435–2463, 1999.

[22] Niels Provos. Honeyd-a virtual honeypot daemon. In *10th DFN-CERT Workshop, Hamburg, Germany*, volume 2, 2003.

[23] George F Riley. The georgia tech network simulator. In *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research*, pages 5–12. ACM, 2003.

[24] Haakon Ringberg, Matthew Roughan, and Jennifer Rexford. The need for simulation in evaluating anomaly detectors. *ACM SIGCOMM Computer Communication Review*, 38(1):55–59, 2008.

[25] Christian Rossow, Christian J Dietrich, Chris Grier, Christian Kreibich, Vern Paxson, Norbert Pohlmann, Herbert Bos, and Maarten van Steen. Prudent practices for designing malware experiments: Status quo and outlook. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 65–79. IEEE, 2012.

[26] Colleen Shannon and David Moore. The spread of the witty worm. *Security & Privacy, IEEE*, 2(4):46–50, 2004.

[27] Douglas C Sicker, Paul Ohm, and Dirk Grunwald. Legal issues surrounding monitoring during network research. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 141–148. ACM, 2007.

[28] Arno Wagner and Bernhard Plattner. Entropy based worm and anomaly detection in fast ip networks. In *Enabling Technologies: Infrastructure for Collaborative Enterprise, 2005. 14th IEEE International Workshops on*, pages 172–177. IEEE, 2005.

[29] Michele C Weigle, Prashanth Adurthi, Félix Hernández-Campos, Kevin Jeffay, and F Donelson Smith. Tmix: a tool for generating realistic tcp application workloads in ns-2. *ACM SIGCOMM Computer Communication Review*, 36(3):65–76, 2006.

[30] Vinod Yegneswaran, Paul Barford, and Somesh Jha. Global intrusion detection in the domino overlay system. In *Proceedings of NDSS*, volume 2004. San Diego, CA, 2004.

[31] Jian Zhang, Phillip Porras, and Johannes Ullrich. Highly predictive blacklisting. In *USENIX Security*, volume 8, pages 107–122, 2008.