Outsourcing Home Network Security

Nick Feamster
School of Computer Science, Georgia Tech

ABSTRACT

The growth of home and small enterprise networks brings with it a large number of devices and networks that are either managed poorly or not at all. Hosts on these networks may become compromised and become sources of spam, denial-of-service traffic, or the site of a scam or phishing attack site. Although a typical user now knows how to apply software updates and run anti-virus software, these techniques still require user vigilance, and they offer no recourse when a machine ultimately becomes compromised. *Rather than having individual networks managed independently, we propose to outsource the management and operation of these networks to a third party that has both operations expertise and a broader view of network activity.* Our approach harnesses two trends: (1) the advent of programmable network switches, which offer flexibility and the possibility for remote management; and (2) the increasing application of distributed network monitoring and inference algorithms to network security problems (an appealing technique because of its ability to reveal coordinated behavior that may represent an attack).

Categories and Subject Descriptors: C.2.0 [Computer-Communication Networks]: Security and Protection C.2.3 [Computer-Communication Networks]: Network Management C.2.6 [Computer-Communication Networks]: Internetworking

General Terms: Algorithms, Design, Management, Reliability, Security

Keywords: home networking, network security, programmable networks

1. Introduction

An increasing number of network devices are connected to the Internet through small networks in the home or small enterprise. As with larger enterprise and carrier networks, these smaller networks also require continued administration both to contain attacks and maintain high availability in the case of network failures. Unfortunately, today's networks are difficult to operate, manage, and secure; smaller home and enterprise networks typically lack the resources to devote to network operations tasks. The Yankee Group has estimated that as much as half of all costs of running a network are operational [21]. Home networks pose a particular challenge to network security because the they are often either managed poorly or not at all. As a result, hosts on these networks often become compromised and become sources of spam, denial-of-service traffic, or sites of a scam or phishing site. Unfortunately, users or operators of these networks typically lack the expertise and vigilance to secure their networks, essentially putting home networks at the center of the ongoing battle for the Internet's overall security.

Users and operators alike have continually called for networks that are easier to manage and more secure; the need for networks that are easy to manage is particularly acute in homes and small enterprise networks, where the fixed cost of having dedicated system administrators may be prohibitively high. One possible approach is to hire expert network administrators to manage the security every small enterprise and home network. Unfortunately, such an approach would not scale, nor would it be affordable. Worse yet, this approach might also be ineffective, because many network security techniques require having a view of network activity from many distinct vantage points. Another approach would be to make individual networks "easier to manage". Past research has explored how to make individual networks more manageable but still presumes that each network is operated by local operators and administrators; home networks and small enterprises may lack any operators whatsoever.

We believe that users and operators of small networks should not be burdened with complex security and management tasks at all. These networks should, to the extent possible, operate in a "plug in and forget" fashion. Towards this vision, this position paper poses an alternate approach: *Rather than having individual networks managed independently, outsource the management and operation of these networks to a third party that has both operations expertise and a broader view of network activity.* Our approach harnesses two trends: (1) the advent of programmable network switches, which offer flexibility and the possibility for remote management; and (2) the increasing application of distributed network monitoring and inference algorithms to network security problems (an appealing technique because of its ability to reveal coordinated behavior that may represent an attack). Each home network would have a programmable gateway that collects statistics about network activity that can serve as input to spam filtering and botnet detection algorithms; sends these aggregate statistics to a controller that performs distributed inference; and takes the appropriate actions to help secure the network at large (*e.g.*, pushing out new filtering rules, IP blacklists, etc.). Although we believe that the potential applications of such an architecture are many, in this paper, we focus on applying this model to

HomeNets 2010 September 3, 2010, New Delhi, India.

the most pressing security problems: spam filtering and botnet detection.

Our proposed approach poses several new, challenging research problems; this paper merely raises these concerns and presents some initial ideas for how we might cope with them. First, the *scale* of data that can be collected in such a system is massive: to effectively detect and correct network security problems with distributed inference, the proposed system must intelligently downsample and aggregate network data without eradicating the information that is necessary to perform diagnosis or detection. Second, our approach poses significant *privacy* challenges because the distributed inference algorithms require collecting sensitive information across networks. Third, the proposed system introduces network devices that must be *remotely managed*; a third party must be able to remotely correct failures and misconfigurations. Finally, the network devices and the controller must be *resilient to attack*; attackers might attempt to evade the distributed detection algorithms or attack the infrastructure itself. Our research agenda will tackle these challenges, as well as additional specific research challenges for each of the distributed inference techniques and will culminate with the deployment of a prototype system on a campus network and several home networks.

2. Design

We now describe an overview of the high-level design for outsourcing network security, as well as the basic building blocks of the system.

2.1 Overview

Previous work, including our own, has proposed using distributed inference to assist operators with network performance and security problems because of its ability to automatically recognize coordinated behavior and traffic patterns. These solutions, however, do not investigate how the results of these inference algorithms might ultimately be used to automate the network operation itself. We propose to *couple distributed inference with network programmability* to automate various network operations tasks that have continually vexed both users and operators. Recent trends in both research and industry have resulted in the development of network switches that are programmable by an external controller via a standardized interface [25].

We propose to equip each independently operated network with programmable switches that can (1) collect statistics about network activity in the wide area; (2) send aggregate network statistics to a controller that performs distributed inference to determine sources of unwanted network traffic; (3) change the way they forward specific traffic flows based on commands from a centralized controller. Although we believe that the potential applications of such an architecture are many, in this proposal, we focus on applying this model to the reduction of unwanted traffic (*e.g.*, botnets, spam).

Figure 1 presents an overview of the system. Each local network contains one or more programmable *switches* that are instrumented with the standardized programmable interface [25]. These switches collect information about local network activity, including (1) DNS lookup patterns of local hosts on the network; (2) inbound and outbound email activity; (3) response times to various external destinations. The switches aggregate this information and send it to one or more third-party *controllers*.

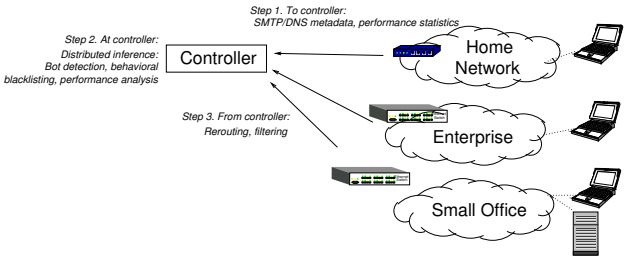

Figure 1: System overview. Programmable switches in homes and small enterprises pass metadata to a controller, which performs distributed inference to detect sources of unwanted traffic (*e.g.*, bots, spam senders). After detection, the controller may use a standard API to install flow table entries in switches to filter unwanted traffic.

A centralized *controller* then applies distributed inference algorithms to detect performance and security problems on these local networks and sends commands back to the switches to control how traffic leaves the networks. The controller can infer that hosts on its network may be members of a botnet and install filters on the local switch to block botnet traffic. The controller can also identify spammers—either on networks where the switches are deployed or in the wide area—and install filters on the switches to prevent hosts on the local networks from either sending or receiving spam traffic. The system has two main components: (1) distributed network monitoring and inference (*i.e.*, the algorithms that detect security or performance problems); and (2) control, via programmable network elements. The rest of the section describes these components.

2.2 System Components

Distributed network monitoring. Network elements in each home network act as "sensors", gathering security-related network data (*e.g.*, DNS lookups, attack traffic) that reflect activity on the local network. These elements may periodically aggregate this data and either exchange it with similar elements in other networks or pass this data to a central processing node. These elements may collect data passively, and also initiate active measurements at the request of a third-party controller.

Distributed inference and data mining. Network elements that are managed by a third party (i.e., not the operators of the networks themselves) aggregate data from sensors and perform distributed inference. The nature of this inference depends on the specific network management problem. We consider the function of inference in the case of automatically stopping the flow of unwanted traffic. Small networks such as home networks and small offices need to protect themselves against unwanted traffic, such as spam and phishing attacks, but these networks are also increasingly becoming the source of unwanted traffic. Our recent work has demonstrated that applying distributed inference to cluster both control and attack traffic from compromised

hosts can lead to accurate, automated containment of botnet-related activities (*e.g.*, spam, phishing) [15, 27, 32]. We believe that deploying programmable switches that also serve as sensors will assist us in collecting data from a wide range of networks, as is often needed for distributed inference.

Programmable network elements. The output of distributed inference is most useful if it can be "pushed back into the network" to automatically mitigate various security problems. For this purpose, we exploit the programmability of the network elements deployed in home networks and small enterprises: The third-party elements that perform distributed inference will compute information that can help these switches automatically filter unwanted traffic or circumvent performance problems. A third party might take certain actions such as installing specific filters on the switch (to prevent certain traffic from entering or leaving the network), or possibly even alternate routing table entries (e.g., encapsulating traffic to automatically re-route it around a performance problem).

3. Application: Home Network Security

Our design should amortize the network management burden for operations tasks ranging from troubleshooting to security. This paper focuses in particular on two important security tasks where we have already developed inference algorithms that might be distributed across programmable home network devices: spam filtering, and detection of botnets and malware.

3.1 Spam Filtering

Our previous work applies distributed monitoring to collect network-level features to learn *behavioral fingerprints* that distinguish spammers from legitimate email senders [15, 32]. We briefly summarize our ongoing work in developing these algorithms, our proposed work to integrate these algorithms into a distributed inference system, as well as the associated research challenges. For example, SpamTracker [32] and SNARE [15] are distributed, real-time algorithms for establishing the reputation of email senders. They take input from a wide variety of distributed mail servers, detect coordinated email sending activity that resembles spamming activity, and automatically install filters that quarantine spam.

We could incorporate our classification algorithms into a distributed inference system that (1) takes input from switches in local networks and (2) pushes appropriate flow table entries (*i.e.*, to block or redirect traffic) into these switches to stop spam traffic as close as possible to its source. SpamSpotter builds classifiers by aggregating features across email sending activity collected from local networks and computing behavioral fingerprints that represent likely spam activity. These fingerprints—which take the form of clusters or decision trees—can be pushed to the local switches themselves and used to block IP addresses on local networks as soon as a host exhibits email sending behavior that resembles that of known spammers.

3.2 Botnet and Malware Detection

BotMiner [14] is a botnet-detection algorithm that analyzes network traffic and clusters similar communication activities in the *C-plane* (command and control, or C&C, communication traffic), clusters similar malicious activities in the *A-plane* (activity traffic), and performs cross cluster correlation to identify the hosts that share both similar communication patterns *and* similar malicious activity patterns. These hosts, by definition, are bots in the monitored network. We have also developed malware detection algorithms for local area networks that examine network traffic originated from or received by the hosts of a network and apply both signature and behavioral analysis to identify the (groups of) hosts whose behavior match models or definitions of botnets [27].

Although these algorithms were designed for an enterprise setting, the crux of the approach—clustering flow statistics to identify coordinated, bot-like behavior—could apply in a distributed botnet detection and response system that (1) collects traffic data on suspicious (botnet-like) communication and activities from local network switches, (2) identifies botnets and hence their C&C and activity servers using the aggregate data, and (3) pushes the list of domain names of these botnet servers to the switches to block bots from accessing them. We envision that home networks are an ideal setting for such an application: each home networking device could collect network flow statistics from individual homes, aggregate and cluster those flow statistics to detect coordinated activity, and subsequently take corrective action (e.g., filtering) within the respective home networks.

4. Research Challenges

Our approach poses several new, challenging research problems. First, the *scale* of data that can be collected in such a system is massive: to effectively detect and correct network security problems with distributed inference, the proposed system must intelligently downsample and aggregate network data without eradicating the information that is necessary to perform diagnosis or detection. Second, our approach poses significant *privacy* concerns, because the distributed inference algorithms require collecting sensitive information across networks. Third, the proposed system introduces network devices that must be *remotely managed*; a third party must be able to remotely correct failures and misconfigurations. Fourth, the network devices and the controller must be *resilient to attack*; attackers might attempt to evade the distributed detection algorithms or attack the infrastructure itself. Finally, the system must be able to *resolve policy conflicts* that may between the home network administrator and the central controller; in particular, we must develop mechanisms for resolving questions of whether—and when—a remote third-party controller should be able to override the policy of the local home network. We elaborate on the scalability and privacy challenges below.

4.1 Scalability

The system must cope with scale: given the massive number of small networks, a third party will not be able to col-

lect, aggregate, and analyze a large amount of traffic from each network. Thus, determining how to appropriately sample network traffic in a way that still permits distributed inference is an important open problem. Along these lines, new distributed inference algorithms may be needed to cope with the large, distributed nature of this network data; some algorithms might even be "closed-loop", asking for additional network data from devices on-demand, rather than collecting all data a priori.

Several techniques may help the system scale. One possibility might be to intelligently sample the data collected from the gateway devices in homes. For example, home networks may have similar configurations, devices, or access-link technologies; rather than collecting data from every home gateway, the system might sample traffic data from a subset of networks in a cluster that share common features (and, hence, might share vulnerabilities). For the case of spam filtering algorithms that require labeled input for training, our preliminary experiments show that a spam classifier can be extremely accurate, even with a random sample of a few thousand legitimate emails and spam messages per day. Another possibility is to send very little data. Another approach that may work for certain applications is to send only a small amount of data to a central controller initially (e.g., sampled flow summaries, or alerts about specific events), but give the controller the ability to collect more data from the gateways in home network on demand. A key challenge in this regard will be to design multi-stage detection algorithms that can initially operate on much more coarse-grained data but can subsequently gather more copious, fine-grained information from home networks when the need arises.

4.2 Privacy

The proposed distributed inference algorithms require edge networks to export and share data about the traffic on their networks. As part of the proposed architecture, we must determine the data that each network needs to expose to make these management problems easier, without compromising privacy. Users may exhibit spammer- or bot-like behaviors when they visit questionable Web sites, inadvertently download malicious programs and command data, or send certain emails. Thus, when a local switch uploads local traffic data, it needs to anonymize or aggregate potentially sensitive information.

Despite the significant privacy challenges that this system poses, there are at least two possible directions forward. First, the algorithms that we have developed for the applications described in Section 3 do not require any payload information; they only require flow-level statistics. In some cases, however, even the information in flow-level statistics (e.g., IP addresses) may be sensitive. Hence, one possible direction to explore might be to develop a new class of anomaly detection algorithms that can operate at the granularity of IP prefixes, rather than individual IP addresses. Home gateways could then obfuscate the IP addresses in their flow statistics; if the global detection algorithm detected an anomaly, it could then notify the Internet service provider corresponding to the IP prefix, who could then investigate the behavior of individual IP addresses. The main challenge with such an approach would be to determine whether and to what extent prefix-level anonymization would aggregate flow statistics in a way that obfuscated traffic anomalies that would otherwise be visible if statistics were gathered at the granularity of IP addresses. Second, privacy-preserving algorithms that perform some analysis locally before uploading statistics to a central server, and obscuring IP addresses whenever possible, may help protect user privacy; existing algorithms such as SEPIA [3] may be applicable.

5. Deployment Plans

To control the network elements, we plan to use the OpenFlow switch specification [2], which allows a remote control element to remotely install flow table entries in switches via a secure channel and standard OpenFlow API commands. OpenFlow switches can install flow table entries that take one of four actions: (1) *forward* packet to a certain output port; (2) *encapsulate* a packet to forward it to an alternate destination; (3) *drop* the packet; (4) *send* the packet along the normal processing pipeline.

Figure 1 shows the general approach; Step 3 shows how the controller might install state in the switches to take corrective action (*i.e.*, filter unwanted traffic, redirect traffic along working paths). We plan to build on both the Linux reference implementation of the OpenFlow switch, and the NOX Box platform [1] as the basis for a switch that we could ultimately deploy in home networks and small enterprises.

6. Related Work

We describe related work in network security and distributed inference for network monitoring.

6.1 Security

Behavioral modeling of email sending patterns. Our detection algorithms rely on the ability to observe coordinated activity across email senders that is characteristic of a botnet mounting a spam campaign, which bears similarity to various previous work on behavioral modeling of email sending behavior. Hershkop *et al.* suggested techniques for analyzing email by looking at behavioral features of users (*e.g.*, sending patterns of individual users), as well as n-gram analysis and keyword spotting [16, 36]. Recent work studied the properties of spam campaigns using various monitoring techniques, including setting up open proxies [26] and by infiltrating the spamming botnet itself [8]. Both of these studies also observe coordinated spamming behavior that we aim to detect automatically.

Spam filtering using network-level features. Recent years have seen work that builds on our early study of the *network-level behavior* of spammers. Our work studied the network-level behavior of spammers, with an eye towards developing filters that are based on behavioral features (*i.e.*, how the spam was sent, as opposed to the contents of individual messages) [31]. Clayton *et al.*'s spamHINTS project has also recently been developing techniques for distinguishing spammers from legitimate senders [9]. Xie *et al.* [39] discov-

ered that a vast majority of mail servers running on dynamic IP address were used solely to send spam; they also recently presented a technique to automatically identify bots by using signatures constructed from URLs in spam messages [38]. Beverly and Sollins built a sender-reputation classifier based on transport-level characteristics (*e.g.*, round-trip times, congestion windows) [7] using a support vector machine. Other work has also attempted to group senders based on recipient [13, 19, 24].

Sender blacklists. Existing blacklists (*e.g.*, Spamhaus [35], SpamCop [34]) maintain sender reputation according to senders' IP addresses. Although these blacklists generally have low false positive rates, our previous work has demonstrated that blacklists are both incomplete (*i.e.*, they have a low detection rate) and unresponsive (*i.e.*, they list active spamming IP addresses sometimes as long as months after they first become active) [30, 32]. IronPort [18] and Secure Computing [33] offer spam filtering appliances that could ultimately incorporate various reputation algorithms.

6.2 Distributed Inference

Data sharing for network security. The distributed inference detection strategy is motivated by the fact that coordinated attacks may not be visible at any single vantage point but may become visible when viewed from a large collection of vantage points [20]. Allman *et al.* suggested a design for cross-organizational information sharing to improve visibility into coordinated network threats and proposed a private matching mechanism to help distinct monitoring locations share information about network events while still preserving privacy [4, 5]; our work could be viewed as an instance of the collaborative system they envisioned. Existing collaborative filtering systems examine message contents, as reported or submitted by users or mail servers [10, 12, 22, 28, 29, 37], as opposed to *network-level properties*. Other work has examined how to scalably aggregate monitoring information collected in darknet traffic for botnet or attack monitoring [6, 11] and showed that source-based sampling can be an effective data reduction technique. Some of these sampling techniques might prove useful for improving scalability.

Distributed anomaly detection. Recent work on distributed inference has augmented centralized traffic anomaly detection algorithms (*e.g.*, [23]) by performing local classification based on approximate measurements and passing updates to a central coordinator when a local node deems that its local measurements deviate from the estimates that the central coordinator is using as the basis for training [17]. These techniques can be used when mail servers perform chiefly local classification with occasional updates from other nodes.

7. Summary and Research Agenda

An increasing number of devices that are connected to the Internet lie in home networks. The emergence of these networks essentially makes every user an unskilled network operator. In light of the rise of botnets—and the prevalence of bots within home networks—it is critical that we find a way to secure these networks. Unfortunately, we cannot rely on unskilled home network users to do so. As a possible way forward, we have proposed that users *outsource* security-related home network management tasks to an off-site controller that can both detect Internet-wide coordinated activity coming from homes and automatically take corrective action on behalf of these home users, via controllable network gateways. Of course, the list of challenges towards realizing this type of outsourcing is daunting; most notably, striking the appropriate balance between user privacy and security will be difficult, but we do believe that an appropriate balance exists, and that our current algorithms that are based only on high-level network traffic statistics (rather than payloads) could serve as a useful starting point.

Acknowledgments

This work was funded by NSF CAREER Award CNS-0643974 and NSF award CNS-0721581. We thank Ellen Zegura and the anonymous reviewers for helpful comments on earlier versions of this paper.

REFERENCES

[1] NOX Box. `http://noxrepo.org/manual/noxbox.html`.

[2] OpenFlow Specification v0.8.2. `http://yuba.stanford.edu/openflow/documents/openflow-spec-v0.8.2.pdf`.

[3] SEPIA: Aggregation of Network Measurements Using Multiparty Computation. In *Passive & Active Measurement (PAM)*, Zurich, Switzerland, Apr. 2010.

[4] M. Allman, E. Blanton, V. Paxson, and S. Shenker. Fighting Coordinated Attackers with Cross-Organizational Information Sharing. In *Proc. 5th ACM Workshop on Hot Topics in Networks (Hotnets-V)*, Irvine, CA, Nov. 2006.

[5] M. Allman, C. Kreibich, , V. Paxson, R. Sommer, and N. Weaver. Principles for Developing Comprehensive Network Visibility. In *Proc. 3rd Usenix Workshop on Hot Topics in Security (HotSec)*, San Jose, CA, July 2008.

[6] M. Bailey, E. Cooke, F. Jahanian, N. Provos, K. Rosaen, and D. Watson. Data Reduction for the Scalable Automated Analysis of Distributed Darknet Traffic. In *Proc. ACM SIGCOMM Internet Measurement Conference*, New Orleans, LA, Oct. 2005.

[7] R. Beverly and K. Sollins. Exploiting the Transport-Level Characteristics of Spam. In *5th Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 2008.

[8] Chris Kanich and Christian Kreibich and Kirill Levchenko and Brandon Enright and Vern Paxson and Geoffrey M. Voelker and Stefan Savage,. Spamalytics: an Empirical Analysis of Spam Marketing Conversion. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, Arlington, VA, Oct. 2008.

[9] R. Clayton. spamHINTS: Happily It's Not The Same. `http://www.spamhints.org/`, 2007.

[10] Cloudmark Authority Anti-Spam. `http://www.cloudmark.com/serviceproviders/authority/spam/`.

[11] E. Cooke, F. Jahanian, and D. McPherson. The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets. In *1st USENIX Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, Cambridge, MA, July 2005.

[12] E. Damiani, S. de Vimercati, and P. Samarati. P2P-Based Collaborative Spam Detection and Filtering. In *4th IEEE Conference on P2P*, 2004.

[13] L. H. Gomes, F. D. O. Castro, R. B. Almeida, L. M. A. Bettencourt, V. A. F. Almeida, and J. M. Almeida. Improving Spam Detection Based on Structural Similarity. In *Proc. SRUTI Workshop*, Cambridge, MA, July 2005.

[14] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *Proc. 17th USENIX Security Symposium*, Vancouver, BC, Canada, Aug. 2008.

[15] S. Hao, N. Syed, N. Feamster, A. Gray, and S. Krasser. Detecting Spammers with SNARE: Spatio-temporal Network-level Automatic Reputation Engine. In *Proc. 18th USENIX Security Symposium*, Montreal, Quebec, Canada, Aug. 2009.

[16] S. Hershkop. *Behavior-based Email Analysis with Application to Spam Detection*. PhD thesis, Columbia University, 2006.

[17] Y. Huang, N. Feamster, A. Lakhina, and J. Xu. Exposing routing problems with network-wide analysis. Technical report, Georgia Tech, May 2006. Number forthcoming. Available upon request.

[18] IronPort Carrier Grade Email Security Appliance. `http://www.ironport.com/products/ironport_x1000.html`, 2007.

[19] L. Johansen, M. Rowell, K. Butler, and P. McDaniel. Email Communities of Interest. In *4th Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 2007.

[20] S. Katti, B. Krishnamurthy, , and D. Katabi. Collaborating Against Common Enemies. In *Proc. ACM SIGCOMM Internet Measurement Conference*, New Orleans, LA, Oct. 2005.

[21] Z. Kerravala. Configuration management delivers resiliency. Technical report, The Yankee Group, Nov. 2002.

[22] J. Kong et al. Scalable and Reliabile Collaborative Spam Filters: Harnessing the Global Socail Email Networks. In *3rd Annual Workshop on the Weblogging Ecosystem*, 2006.

[23] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft. Structural analysis of network traffic flows. In *Proc. ACM SIGMETRICS*, pages 61–72, New York, NY, June 2004.

[24] H. Lam and D. Yeung. A learning approach to spam detection based on social networks. In *4th Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 2007.

[25] OpenFlow Switch Consortium. `http://www.openflowswitch.org/`, 2008.

[26] A. Pathak, Y. C. Hu, and Z. M. Mao. Peeking into Spammer Behavior from a Unique Vantage Point. In *Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, San Francisco, CA, Apr. 2008.

[27] R. Perdisci, W. Lee, and N. Feamster. Behavioral Clustering of HTTP-Based Malware. In *Proc. 7th USENIX NSDI*, San Jose, CA, Apr. 2010.

[28] V. Prakash. Vipul's Razor. `http://razor.sourceforge.net/`, 2007.

[29] Pyzor. `http://pyzor.sourceforge.net/`.

[30] A. Ramachandran, D. Dagon, and N. Feamster. Can DNSBLs Keep Up with Bots? In *3rd Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, July 2006.

[31] A. Ramachandran and N. Feamster. Understanding the Network-Level Behavior of Spammers. In *Proc. ACM SIGCOMM*, Pisa, Italy, Aug. 2006. An earlier version appeared as Georgia Tech TR GT-CSS-2006-001.

[32] A. Ramachandran, N. Feamster, and S. Vempala. Filtering spam with behavioral blacklisting. In *Proc. 14th ACM Conference on Computer and Communications Security*, Alexandria, VA, Oct. 2007.

[33] Secure Computing IronMail. `http://www.securecomputing.com/index.cfm?skey=1612`, 2007.

[34] SpamCop. `http://www.spamcop.net/`.

[35] Spamhaus, 2006. `http://www.spamhaus.org/`.

[36] S. J. Stolfo, S. Hershkop, C.-W. Hu, W.-J. Li, O. Nimeskern, and K. Wang. Behavior-based modeling and its application to Email analysis. 6(2):187–221, May 2006.

[37] P. Vixie. Distributed Checksum Clearinghouse. `http://www.rhyolite.com/anti-spam/dcc/`.

[38] Y. Xie, F. Yu, , K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming bots: Signatures and characteristics. In *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008.

[39] Y. Xie, F. Yu, K. Achan, E. Gilum, M. Goldszmidt, and T. Wobber. How dynamic are IP addresses. In *Proc. ACM SIGCOMM*, Kyoto, Japan, Aug. 2007.