UNIVERSITY OF CALIFORNIA, SAN DIEGO

Characterizing Internet Scams through Underground Infrastructure Infiltration

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Christopher William Kanich

Committee in charge:

Professor Stefan Savage, Co-Chair
Professor Geoffrey M. Voelker, Co-Chair
Professor Rene L. Cruz
Professor Tara Javidi
Professor Lawrence K. Saul

2012

The dissertation of Christopher William Kanich is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____
Co-Chair

_____
Co-Chair

University of California, San Diego

2012

DEDICATION

To my family.

EPIGRAPH

You follow drugs, you get drug addicts and drug dealers. But you start to follow the money, and you don't know where the fuck it's gonna take you.

—Lester Freamon

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am extremely thankful for the guidance of my advisors, Professors Stefan Savage and Geoffrey M. Voelker. Grad school is a voyange of discovery: above all else, I have discovered the amount of care and effort they put into everything the do. Words simply cannot express how in awe and thankful I am to have the privilege of being advised by them over the past seven years.

Beyond my advisors, I absolutely would not have made it this far without the enthusiastic assistance of my collaborators: Vern Paxson, Christian Kreibich, Chris Grier, Nick Weaver, Weidong Cui, and Márk Félegyházi, Kirill Levchenko, Brandon Enright, Tristan Halvorson, He 'Lonnie' Liu, Damon McCoy, Marti Motoyama, Andreas Pitsillidis, Stephen Checkoway, and David Wang. Research is a collaborative effort and these collaborators have provided valuable insight and helped me enjoy the process.

The UCSD CSE department's vivacious grad students have made coming in to the office every day a pleasure. I've also received valuable feedback and assistance from John McCullough, Cynthia Taylor, Michael Vrable, Kai Wang, Jeannie Albrecht, Yu-Chung Cheng, Ramana Kompella, Marvin McNett, Ding Yuan, Xiao Ma, Barath Raghavan, Calvin Hubble, and Patrick Verkaik.

I thank my thesis committee members Lawrence K. Saul, Tara Javidi, and Rene Cruz for their valuable feedback and support from proposal to completion of this effort.

Finally, I thank my family.

Chapters 1, 2, and 3, in part, are a reprint of the material as it appears in Proceedings of ACM CCS 2008. Kanich, Chris; Kreibich, Christian; Levchenko, Kirill; Enright, Brandon; Voelker, Geoffrey M.; Paxson, Vern; Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapters 1, 2, and 4, in part, are a reprint of the material as it appears in Proceedings of USENIX Security 2011. Kanich, Chris; Weaver, Nicholas; McCoy, Damon; Halvorson, Tristan; Kreibich, Christian; Levchenko, Kirill; Paxson, Vern; Voelker, Geoffrey M., Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, is a reprint of the material as it appears in Proceedings of IMC 2011. Kreibich, Christian, Weaver, Nicholas; Kanich, Chris; Cui, Weidong; Paxson, Vern.

Chapter 4, in part, is a reprint of the material as it appears in Proceedings of USENIX CSET 2011. Kanich, Chris; Chachra, Neha; McCoy, Damon; Grier, Chris; Wang, David; Motoyama, Marti; Levchenko, Kirill; Savage, Stefan; Voelker, Geoffrey M. The dissertation author was the primary investigator and author of this paper.

VITA

2005	B.S. in Mathematics and Computer Science Purdue University West Lafayette
2012	Ph.D. in Computer Science, University of California, San Diego

PUBLICATIONS

Vaibhav Garg, Chris Kanich, and L. Jean Camp. " Analysis of E-Crime in Crowd-sourced Labor Markets: Mechanical Turk vs. Freelancer." In *Proceedings of the Workshop on the Economics of Information Security*, Berlin, Germany, 2012.

Christian Kreibich, Nicholas Weaver, Chris Kanich, Weidong Cui, and Vern Paxson. "GQ: Practical Containment for Measuring Modern Malware Systems." In *Proceedings of the ACM Internet Measurement Conference*, Berlin, Germany, 2011.

Chris Kanich, Nicholas Weaver, Damon McCoy, Tristan Halvorson, Christian Kreibich, Kirill Levchenko, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. "Show Me the Money: Characterizing Spam-advertised Revenue." In *Proceedings of the 20th USENIX Security Symposium*, San Francisco, California, USA, 2011.

Chris Kanich, Stephen Checkoway, and Keaton Mowery. "Putting out a HIT: Crowdsourcing Malware Installs." In *Proceedings of the 5th USENIX Workshop on Offensive Technologies*, San Francisco, California, USA, 2011.

Chris Kanich, Neha Chachra, Damon McCoy, Chris Grier, David Wang, Marti Motoyama, Kirill Levchenko, Stefan Savage, and Geoffrey M. Voelker. "No Plan Survives Contact: Experience with Cybercrime Measurement." In *Proceedings of the 4th USENIX Workshop on Cyber Security Experimentation and Test*, San Francisco, California, USA, 2011.

Kirill Levchenko, Andreas Pitsillidis, Neha Chachra, Brandon Enright, Márk Félegyházi, Chris Grier, Tristan Halvorson, Chris Kanich, Christian Kreibich, He Liu, Damon McCoy, Nicholas Weaver, Vern Paxson, Geoffrey M. Voelker, and Stefan Savage. "Click Trajectories: End-to-End Analysis of the Spam Value Chain." In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, California, USA, 2011.

Marti Motoyama, Kirill Levchenko, Chris Kanich, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. "Re: CAPTCHAs – Understanding CAPTCHA Solving from an Economic Context." In *Proceedings of the 19th USENIX Security Symposium*, Washington, DC, USA, 2010.

Andreas Pitsillidis, Kirill Levchenko, Christian Kreibich, Chris Kanich, Geoffrey M. Voelker, Vern Paxson, Nicholas Weaver, and Stefan Savage. "Botnet Judo: Fighting Spam with Itself." In *Proceedings of The Network & Distributed System Security Conference*, San Diego, CA, USA, 2010.

Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. "Spamalytics: An Empirical Analysis of Spam Marketing Conversion." In *ACM Communications of the Association for Computing Machinery*, pp. 99–107, 2009.

Christian Kreibich, Chris Kanich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. "Spamcraft: An Inside Look At Spam Campaign Orchestration." In *Proceedings of the Second USENIX Workshop on Large-Scale Exploits and Emergent Threats*, Boston, MA, USA, 2009.

Christian Kreibich, Chris Kanich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. "On the Spam Campaign Trail." In *Proceedings of the First USENIX Workshop on Large-Scale Exploits and Emergent Threats*, San Francisco, CA, 2008.

Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. "Spamalytics: An Empirical Analysis of Spam Marketing Conversion." In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, Alexandria, VA, USA, 2008.

Chris Kanich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, and Stefan Savage. "The Heisenbot Uncertainty Problem: Challenges in Separating Bots from Chaff." In *First USENIX Workshop on Large-Scale Exploits and Emergent Threats*, San Francisco, CA, USA, 2008.

Brandon Enright, Geoff Voelker, Stefan Savage, Chris Kanich, and Kirill Levchenko. "Storm: When Researchers Collide." In *;login: The USENIX Magazine*, pp. 6–13, 2008.

ABSTRACT OF THE DISSERTATION

Characterizing Internet Scams through Underground Infrastructure Infiltration

by

Christopher William Kanich

Doctor of Philosophy in Computer Science

University of California, San Diego, 2012

Professor Stefan Savage, Co-Chair
Professor Geoffrey M. Voelker, Co-Chair

Modern unsolicited bulk email, or spam, is ultimately driven by product sales: goods purchased by customers online. While this model is easy to state in the abstract, our understanding of the concrete business environment—how many orders, of what kind, from which customers, for how much—is poor at best. This situation is unsurprising since such sellers typically operate under questionable legal footing, with ground truth data rarely available to the public. However, absent quantifiable empirical data, "guesstimates" operate unchecked and can distort both policy making and our choice of appropriate technical interventions.

This dissertation presents new methodologies for and results from experiments that characterize and quantify the economics of email based scams. The methodology

relies on *infrastructure infiltration* to gain a view of the mechanisms and revenues of these operations from the point of view of the perpetrators themselves. Through multiple research efforts, we are able to capitalize on the weaknesses of the perpetrators' security to collect information that provides insight into the way these scams work.

The first effort investigates the proportion of spam recipients that act upon the spam messages they receive - the "conversion rate" of spam. Using a parasitic infiltration of an existing botnet's infrastructure, we analyze two spam campaigns comprised of nearly half a billion email messages: one campaign designed to propagate a malware Trojan, the other campaign marketing on-line pharmaceuticals. We identify the number that are successfully delivered, the number that pass through popular anti-spam filters, the number that elicit user visits to the advertised sites, and the number of "sales" and "infections" produced.

The second effort uses two inference techniques to peer inside the business operations of spam-advertised enterprises: purchase pair and basket inference. Using these methodologies, I provide informed estimates on order volumes, product sales distribution, customer makeup and total revenues for a range of spam-advertised businesses.

The results from these studies demonstrate that infiltration of Internet criminal infrastructure allows collection of useful information that can improve our understanding of the operations and economics of adversaries on the Internet. This information informs both technical and policy based defenses so that they can take into consideration the business realities of economically motivated Internet adversaries.

Chapter 1

Introduction

Email is a near-ubiquitous staple of modern life and one of the great success stories of the Internet. Its global reach, infinitesimal cost, and near-instant delivery have transformed communication. Yet these desirable properties have attracted those who wish to profit through abusing the system, including supposed Nigerian princes promising millions of dollars, scammers sending malware purporting to be funny jokes sent from friends, or advertisers hawking pharmaceuticals without a prescription. Billions of these unsolicited "spam emails" are sent every day and are an annoyance to many.

While certainly a nuisance, these unwanted messages cause real harm to email users. For example, one individual lost over five million dollars to an "advance fraud" scheme.[1] While unfortunate, a global and mostly anonymous communication medium will no doubt enable some amount of fraud. However, the spam epidemic not only causes direct financial losses, but causes time, concentration, and productivity to all be lost when end users are interrupted by receiving useless spam messages. This loss itself is only incurred for spam messages that bypass any anti-spam filtering in place: billions of dollars are spent each year on anti-spam products which exist to prevent spam from overwhelming users' inboxes. In a world without billions of spam messages being sent every day, this lost time and money could be better spent elsewhere.

Legitimate users' losses due to spam are harmful, but in many cases users' losses are the perpetrators' gains: these losses fund the operations of abusive users, enabling them to invest in improving their own operations. The sale of these goods is

[1] Adler v. Republic of Nigeria. 219 F.3d 869 (9th Cir. 2000).

also dangerous in many cases, as these spammers are selling pharmaceuticals without a physician's prescription and are not subject to safety regulations. All of these avenues for harm through sending spam share an important property: they all earn money for their perpetrator.

Existing research regarding spam largely focuses on determining what is spam and what is not, and how to prevent spam from being delivered. Approaching spam as a classification problem has been very fruitful, and many of these techniques are presented in Chapter 2. While effective, these techniques only address the symptoms of the problem and not the root cause: spam can be used to make money.

Fighting the root cause can be more effective as a global solution rather than a local one. While local spam filters only protect local users, the incentive to send spam still exists as long as enough potential customers lack adequate filters. To disincentivize spammers, we must look beyond the symptom and evaluate the incentive structure itself: how do spammers make money, where does it come from, and how much do they make? With an understanding of the incentives behind spamming, we can build more effective countermeasures that address the root cause of the problem itself.

As an instrument for making money, spam is equivalent to a direct-mail advertisement, much like a mail-order catalog: advertisements are individually addressed, and the sender achieves profitability when the average net revenue per sale exceeds the cost of sending these advertisements multiplied by the proportion of recipients who make purchases. If we are able to quantify this conversion rate and the magnitudes involved, we can better understand the effectiveness of spam from the spammer's point of view.

Before this research, data regarding the economic effectiveness of spam campaigns was limited to industry estimates, depositions, and anecdotes [74, 1, 80]. Providing a deep, data-driven, and open methodology for measuring the revenues and structure of spam provides a sound footing for effective interventions, policy changes, and new defenses aimed at a global reduction in spam profitability and thus spam volume.

This thesis aims to quantify the questions important to characterizing a spammer's business: who buys their products, how many people click on these advertisements, and how much money do they make? In service of these goals, we capitalize on weaknesses inherent in the business models themselves to opportunistically collect information

regarding the spammers' operations.

1.1 Contributions

I posit that characterizing the economic motivations for spam is possible through infrastructure infiltration and information leakage on the part of the perpetrators. Through infiltration of the Storm botnet, recovery of access logs from a compromised host used to host images for a large pharmaceutical seller, and opportunistic collection of transaction information, we show that this is possible and can lead to fruitful analysis of the Internet's underground economy. The contributions of this dissertation are as follows:

- We successfully investigated the operation and effectiveness of the Storm botnet's operation and spam campaigns. We conducted this study successfully though *sidestepping* the obvious legal and ethical problems associated with sending spam.[2] Critically, this study makes use of an *existing* spamming botnet. By infiltrating its command and control infrastructure parasitically, we convinced it to modify a subset of the spam it *already* sends, thereby directing any interested recipients to servers under our control, rather than those belonging to the spammer. In turn, our servers presented Web sites mimicking those actually hosted by the spammer, but "defanged" to remove functionality that would compromise the victim's system or receive sensitive personal information such as name, address or credit card information.

- We describe a general technique—purchase pair—for estimating the number of orders received (and hence revenue) via on-line store order numbering. We use this approach to establish rough, but well-founded, monthly order volume estimates for many of the leading "affiliate programs" selling counterfeit pharmaceuticals and software.

- We show how to use third-party image hosting data to infer the contents of customer "baskets" and hence characterize purchasing behavior. We apply this technique to a

[2] We conducted our study under the ethical criteria of ensuring *neutral actions* so that users should never be worse off due to our activities, while strictly *reducing harm* for those situations in which user property was at risk.

leading spam-advertised pharmaceutical program and identify both the nature of these purchases and their relation to the geographic distribution of the customer base.

In each case, the real contribution is less in the particular techniques—which an adversary could easily defeat should they seek to do so—but rather in the data that we used them to gather. We document three spam campaigns comprising over 469 million emails. We identified how much of this spam is successfully delivered, how much is filtered by popular anti-spam solutions, and, most importantly, how many users "click-through" to the site being advertised (*response rate*) and how many of those progress to a "sale" or "infection" (*conversion rate*). We also document that seven leading counterfeit pharmacies together have a total monthly order volume in excess of 82,000, while three counterfeit software stores process over 37,000 orders in the same time. Combining advertised prices and demand distributions, we provide an evidence-based estimate that these businesses bring in approximately US$10 million in gross revenue per month.

1.2 Organization

The remainder of this dissertation is organized in the following manner.

Chapter 2 provides background material on spam defenses and related work on the economic approach to cybercrime.

Chapter 3 covers work infiltrating and analyzing the Storm botnet. We explain our methodology for *botnet infiltration* and analyze the information collected by interposing on almost a billion spam messages being sent by this botnet, and the resulting traffic elicited from the recipients.

Chapter 4 explores the demand for and revenue from running an online pharmacy. We infer demand distribution both by customer location and product ordered for over over 750,000 unique visiting IP addresses and over 3,000 distinct purchases. We also describe the *purchase pair* methodology and present the results of inferring the order volume for ten different spam-advertised merchants.

Finally, Chapter 5 concludes, offering a synthesis of these results and discusses future research directions in this space.

Chapter 1, in part, is a reprint of the material as it appears in Proceedings of ACM CCS 2008. Kanich, Chris; Kreibich, Christian; Levchenko, Kirill; Enright, Brandon; Voelker, Geoffrey M.; Paxson, Vern; Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 1, in part, is a reprint of the material as it appears in Proceedings of USENIX Security 2010. Kanich, Chris; Weaver, Nicholas; McCoy, Damon; Halvorson, Tristan; Kreibich, Christian; Levchenko, Kirill; Paxson, Vern; Voelker, Geoffrey M., Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 2

Background and Related Work

Email is one of the great success stories enabled by the Internet; every day, approximately 140 billion messages are exchanged throughout the world [78]. Email is an incredibly cost effective communication medium; all one must do to send an email is know the recipient's address, follow the established protocol, and transmit the message over the Internet to the server in charge of the receiving the recipient's email. There is no postage fee, and no cost to use the protocol. While this is a great boon to those wishing to communicate messages that are useful to both parties, this simplicity is a double edged sword - anyone can send a message to anyone else, and unsolicited, undesired messages are just as easy to send as desired messages.

Since the first reported complaint in 1978, unsolicited bulk email, or spam, has caused untold grief among users, system administrators, and researchers alike. The widespread popularity of email came the onslaught of spam we see today: approximately 78% of all email messages are currently considered spam [76]. The monumental cost effectiveness of email as a communication platform has led to its great success among legitimate senders and spammers alike; the equally monumental annoyance and productivity loss caused by the receipt of unwanted email has itself produced a litany of spam countermeasures.

A fundamental aspect of email that makes spam a hard problem is that in sender-initiated direct communication, there is an information inequality present regardless of medium. While senders understand the usefulness their message will have to the recipient, recipients cannot know a priori that any given message will contain useful information

before expending the effort to find out. This weakness is confounded by the infinitesimal effort necessary to send spam after incurring the startup cost of acquiring a distribution list and Internet connection. Coupled with the anonymous aspect of the protocol itself, these factors make email an incredibly lucrative method of communication for those who have any incentive to contact large numbers of people and don't mind annoying most of them with useless information. When viewed in this light, the spam epidemic becomes almost a foregone conclusion.

SMTP, the Simple Mail Transfer Protocol, is the base protocol used for the transmission of all email worldwide [60]. SMTP was initially designed as a relay mechanism, so that any host could forward any message to any other host, somewhat similar to packet forwarding on the Internet itself. When SMTP was designed in 1982, today's commercial success of the Internet and the pursuant security risks were unimaginable, and no authentication was built into the protocol itself. Even today, many mail servers will accept mail from any IP address attesting to be from any email address. Although this flexibility has its advantages, it also allows many opportunities for abuse.

In response to the deluge of spam, defenders have developed a bevy of defenses aimed at detecting when a message is undesirable, and filtering it out of the user's view, either by discarding it or placing it in a spam subfolder of the user's email account. These classification based mechanisms form a majority of the prior anti-spam research, and have been extensively studied and developed; a survey technical approaches to defeating spam follows in Section 2.1.

Even with a wide array of spam defense mechanisms, spam has continued to exist, and has evolved to circumvent new defenses. Spam defenders developed bayesian spam filters; spammers began to use chaff words and generic messages in order to circumvent content based filtering techniques [58]. Defenders implement blacklists enumerating hosts known to spam; spammers use botnets consisting of millions of hosts to overwhelm the blacklist keepers [63]. Through observing this evolution, we come to realize that fighting spam by filtering undesired messages is fighting a symptom of some deeper phenomenon. Spammers send their messages not to circumvent filtering mechanisms, but because sending spam can make them money. The fact that spam continues to exist and evolve indicates that there is still a healthy return to be made spamming, even in the

face of widely deployed countermeasures. Thus, an alternate approach to spam would be not to attempt to filter messages as one receives them, but instead to disincentivize its senders so that they no longer attempt to send it in the first place. I introduce this economic approach to anti-spam in Section 2.2.

While recognizing the economic nature of this adversary is important, modern spam businesses are realized through complex business processes involving multiple players each specializing in different aspects of the operation. Understanding the structure of the business processes supporting modern spam campaigns is a necessary first step toward finding the correct solutions. I explain the structure of modern spam campaigns in Section 2.3.

An understanding of modern commercial spam campaign structure is helpful, but evidence-based measurements are necessary to place these phenomena into context and motivate creation of the proper economics- and policy-based solutions. I cover the need for these measurements in Section 2.4.

2.1 Traditional Anti-Spam Approaches

Before the success of email and the Internet among the general populace, spam countermeasures consisted of human-manageable whitelists and blacklists for sending IP addresses and mail keywords. The outsize success of the Internet and motivated adversaries changed this landscape and made these countermeasures untenable at scale. Modern spam filtering and prevention can be split into three major categories: content classification, blacklisting, and sender reputation. Each of these strategies builds an anti-spam technique from a different fundamental aspect of spam.

Content classification takes advantage of the fact that, for a given user, the content of undesirable and desirable messages exhibit similarities among themselves and knowledge of previous messages' desirability allows classification of new deliveries; it is covered in Section 2.1.1. Blacklisting capitalizes on the fact that an Internet address which sends unsolicited messages is highly likely to repeat this act, and unlikely to send legitimate, mutually desired communication if blacklisting is in place; it is covered in Section 2.1.2. Sender reputation schemes provide strong ties between the nominal

and true sender of email messages in order to allow records of individual domains' communications to be recorded and used to filter spam based on past behavior; they are covered in Section 2.1.3. Alternative methods not listed above will be briefly covered in Section 2.1.4.

2.1.1 Classification

Content classification is the most popular spam countermeasure in academic literature. The key observation underlying most statistical implementations of this mechanism is that, for a given user, the likely desirability of an unread email message can be estimated based upon its similarity to both previous desired messages and previous undesired messages. Several different machine learning algorithms have been proposed to perform this message clustering task, with Bayesian filtering as the first in 1998 by Sahami et al. [68]. While machine learning based filtering mechanisms usually make decisions locally (either for an individual mail user or organization), some approaches have taken an opposite tack. Rather than judging the desirability of individual tokens and applying this knowledge to newly arriving spam, these approaches determine the desirability of entire messages and store this information globally. One such method is Zhou et al.'s Approximate Object Location Service [88].

Bayesian Junk Email Filtering

Sahami et al. first applied Naive Bayesian classification to the problem of junk mail classification in 1998 [68]. Bayesian classification for spam takes a specific body of text $\mathbf{X}$ as a vector of individual elements (usually words but sometimes as n-grams, all sequences of words or characters of length $|n|$) and learns the overall probability that the message is spam using Bayes theorem for conditional probabilities.

$$P(C = c_k|\mathbf{X} = \mathbf{x}) = \frac{P(\mathbf{X} = \mathbf{x}|C = c_k)P(C = c_k)}{P(\mathbf{X} = \mathbf{x})} \tag{2.1}$$

Bayes Theorem is outlined in Equation 2.1, with C denoting the learned "class" of the message and c_k being the specific class e.g. spam or nonspam, with the generic feature vector $\mathbf{X}$ consisting of all learned words and the elements of $\mathbf{x}$ as constants denoting the presence or absence of a given word within the message being classified.

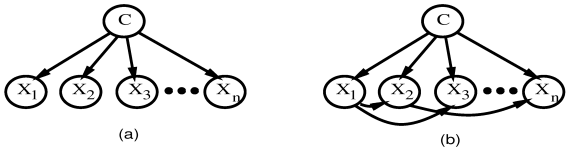


Figure 2.1: Current Bayesian spam filtering uses naive Bayesian networks (a) which assume completely independent probabilities for each predictor, as opposed to complex Bayes (b), which allows multiple levels of interdependence as shown.

Naive Bayesian classification makes the simplifying assumptions that all features of an object are conditionally independent and that only the binary spam/not spam classification is desired. These assumptions allow the computation of Equation 2.1 using Equation 2.2.

$$P(\mathbf{X} = \mathbf{x}|isspam) = \prod_i P(X_i = x_i|isspam) \tag{2.2}$$

In order to apply this classification scheme to email, a learning phase is used to determine the predictive power of each token (x_i from Equation 2.2). With a pre-classified body of emails, the value $P(X_i = x_i|isspam)$ is simply the number of spam messages in which the token x_i appears divided by the total number of messages in which x_i appears. In this paper's implementation, only words with three or more appearances in a corpus of 2,593 messages are included, as the classification certainty of words which appear rarely is low. When evaluating the performance of the classifier against 222 unseen messages which arrived at the same mailbox as the training messages, Naive Bayes exhibited an 8% false positive and 5% false negative rate, with the false positive messages considered relatively less important to the user. The paper also discusses the ability to further classify messages into sub-categories as well as the effect of hand-crafted classifiers on the overall judgement of the anti-spam system.

The application of Naive Bayes learning and classifying to the spam email problem in this paper is an essential contribution to the anti-spam literature. Before Bayesian filtering, spam classification used so-called "domain specific" features of the text which requires the user to manually choose important strings from within emails that signified a message's spam likelihood. While sometimes more flexible than static strings (i.e., an "excessive punctuation" rule which catches any string of 3 or more of the same

punctuation mark), domain specific features are fundamentally limited in their choice and accuracy by being human generated. With the reasonable simplifying assumption that all words in an email can have independent predicting power for the spam likelihood for a given message, Naive Bayes can achieve a consistent 98–99% accuracy rate, even when using a centralized wordlist rather than one tuned to individual email users [6].

Global Message Blacklists

Traditional full message classification systems provide a centralized service advertising raw checksums of messages in order to globally blacklist exact copies of offending messages [66], [61]. Zhou et al introduce a system with two main advances upon this technique, namely *Approximate Text Addressing* (ATA) and *Approximate Distributed Object Location and Routing* (ADOLR) [88]. By combining these complementary capabilities, this system seeks to provide a flexible global storage for records of spam messages in order to halt the spam epidemic itself.

ADOLR builds a distributed object store that uses feature vectors rather than scalar checksums to store the location of a given entity within a traditional distributed object location and routing (DOLR) system such as Pastry [67]. By storing the location of a desired object at the location of every attribute in the feature vector, a client can search for an exact object while only knowing a subset of its attributes. This capability is paired with Approximate Text Addressing, a variant of a technique suggested for finding similar files within a filesystem [51]. ATA provides a deterministically random vector of substring checksums from within a text document. These checksums comprise the feature vector for a given spam email, and are stored into the ADOLR as an approximate record of the spam message's existence. When new messages are received at a site using this system for spam mitigation, the message is hashed with ATA. If enough of these substring checksums are found, the message is considered spam.

The evaluation of ATA presented in [88] highlights the necessity of balancing the similarity threshold between being too strict (causing a low detection rate) and being too liberal (causing false positives). While the authors found suitable settings in their tests, the corpus of only 25,000 emails does not adequately characterize a global spam defense mechanism, as even a single large university as of 2004 was receiving over 45,000 emails

per day [19].

2.1.2 Blacklisting

Although content based classification can be an effective measure against spam, targeting the senders of spam is also a well established countermeasure. Blacklists, in general, are a method of keeping track of bad actors and refusing to take receipt of their email. The main challenge facing an effective and efficient blacklist is maintenance of the bad actor list. This feat has become increasingly difficult in recent years with the proliferation of botnet-sourced spam, which can allow an individual spammer to send one spam campaign from hundreds of thousands of highly transient hosts. Although they face a hefty challenge, modern blacklist techniques enlist a number of novel intuitions and methods to keep their lists of spammers up to date (and, when possible, ahead of) the deluge of unwanted email sent every day.

Blacklisting Research Efforts

Traditionally a blacklist is human maintained or includes addresses automatically in accordance with a policy. Some examples are the Spamhaus PBL blacklist which lists dynamically allocated addresses which should not normally be sending email, and the Spamhaus SBL blacklist which include hosts automatically in response to receiving email at special honeypot domains or addresses [72, 73].

The most common mechanism for implementing a blacklist is through a repurposing of the domain name system [47]. To determine whether an IP address is included on a particular blacklist, one concatenates the IP address with the domain name for the blacklist lookups, and performs a DNS query wishing to determine the IP address for that domain name. Rather than returning the IP address responsible for this domain name, the response IP address encodes existence in the blacklist with a successful lookup, and absence with the “no such domain” response code. This repurposing allows DNS blacklist lookups to be lightweight and efficient through the caching and stateless querying of DNS lookups in addition to not requiring the creation or implementation of a new protocol.

Prior to the first research efforts seeking to improve the effectiveness of DNS Blacklists, Jung and Sit observed that the use of DNSBLs increased greatly between

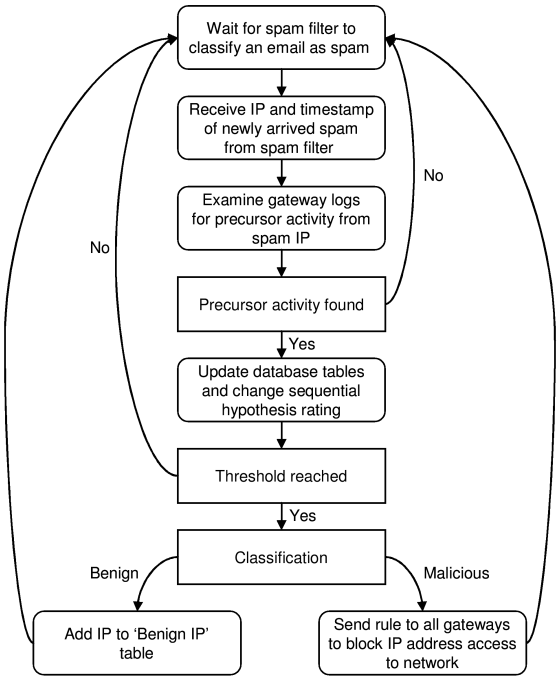


Figure 2.2: The high-level design of the domain specific dynamic blacklist system's online classification phase.

2000 and 2004, and analyzed the blacklisting focus and overlap between several popular blacklists [31]. They suggested that anti-spam systems which query several DNSBLs for each mail received could possibly increase their accuracy by being aware of the overlap between popular blacklists as well as the aggressiveness of each blacklist. In 2006, Ramachandran et al. were able to infiltrate the update mechanism of the Bobax botnet in order to evaluate the coverage of popular DNSBLs for an individual botnet [62]. They found that even though Bobax was exclusively used to send spam and inhabited over two million IP addresses during their 46 day sample, only 4,295 of those hosts were in the Spamhaus DNSBL, a "known to spam" blacklist. Clearly, this leaves room for improvement in the coverage of this blacklist, and motivates the systems outlined below.

Domain Specific Dynamic Blacklists

Cook et al's Domain Specific Dynamic Blacklists combine the techniques of intrusion detection system (IDS) log analysis with blacklists on the mail server in order

to improve the accuracy of both the IDS and the spam filter [10]. The intuition is that spamming hosts will perform anomalous network-level events before spamming the network. By cross referencing the IDS logs with the mail server logs, "precursor activity" from the mail sending IPs can be observed which will increase the likelihood that the sending IP is a spammer; after the likelihood reaches a threshold, the offending IP address is added to the blacklist without human intervention.

This system comprises two phases, one in which log data is filtered by human eyes after restricting the IDS log data to only those hosts sending spam, and one in which rulesets created offline during the first phase are used online to filter new email traffic as it is received. During the first phase, the email server logs and IDS logs are collated for the desired time period, and all IP addresses with no email traffic as well as only email traffic (i.e., no interesting traffic as recorded by the IDS) are discarded. After this phase, human analysis of the logs determines what rules define precursor activity for spamming IPs.

The online portion of the system is outlined in Figure 2.2. The information regarding precursor activity gained during phase 1 is integrated into this feedback loop between the IDS and the spam filter itself. This system actually goes further than a traditional DNSBL by blocking all network traffic from a given spamming IP address rather than only email traffic. The authors note that this system will only blacklist spam after the first message is received, and currently has no expiration policy for entries in the blacklist. However, it is still an important first step in creating accurate and effective automated blacklists.

Behavioral Blacklisting

Ramachandran et al. introduced a system called *SpamTracker* in 2007 that seeks to abstract away content based classification and instead only judge email senders by *how* they send rather than *what* they send [65]. The intuition behind this strategy is that while message content can be easily obfuscated to bypass content filters, network-level features of spammers are more invariant and thus more amenable to being used for blacklisting decisions [63]. The goal of this system is motivated in part by the observations regarding low blacklist coverage for the Bobax botnet seen in [62].

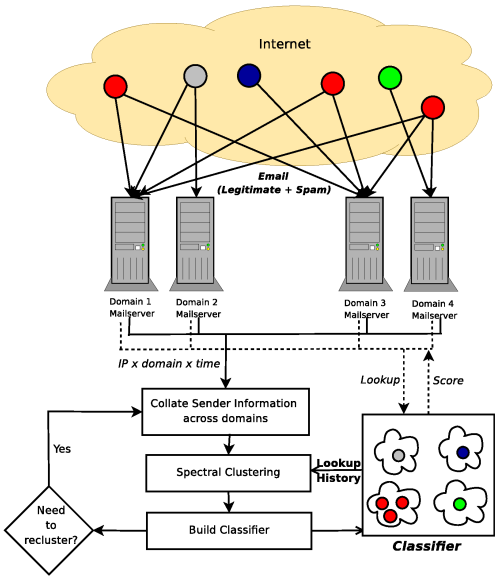


Figure 2.3: The high-level design of *SpamTracker*.

The feature used in *SpamTracker* to identify and cluster spamming hosts is the distribution of destination domains for email sent from a given IP address. The reasoning is that while the distribution of destination domains for any email sender will remain relatively constant within the clustering time frame (6 hours was used in the paper), legitimate senders will not be using a large number of hosts to send to the same distribution of destination domains. Spammers using botnets as their distribution mechanism should exhibit this property, and this is exactly the set of hosts that *SpamTracker* aims to blacklist.

SpamTracker uses a technique called *Spectral Clustering* to group senders by their destination domain distributions in the first stage and then an online classification stage. In the offline clustering stage, the algorithm takes as input logs of spam receipts (with all legitimate reciepts filtered out) at all participating domains. The logs are parsed into arrays of the form (i, j, k) where IP address i sent email at time k to domain j. For a given clustering period, all time values are collapsed and the distribution of all domains j for each individual i are constructed. These distributions are then clustered and labeled with vectors consisting of the destination domain distribution for a given cluster. In the online stage this set of vectors is compared against the distribution for an IP address

sending email to a *SpamTracker*-protected domain, and a similarity metric is computed. If the similarity metric passes a certain threshold, an IP address is considered part of the distributed spamming cluster and blacklisted.

Although *SpamTracker* requires input from several different destination domains to be effective, the results show at least a 10% improvement in the amount of spam caught via blacklisting. The dataset was difficult to fully evaluate as no ground truth data was available for the email or the sending hosts, thus the 10% improvement on the subset of spam that can be captured via DNS blacklists can be considered a lower bound. Additional tuning to find a reasonable threshold setting would be necessary in a production environment. An overview of the *SpamTracker* system is presented in Figure 2.3, displaying the feedback loop between the spam classifier, the spectral clustering algorithm, the participating mailservers and the spammers themselves.

When discussing additional features that could be added, the authors' suggestion of temporal behavior will be very difficult for a spam trap that only sees hundreds or even thousands of destination domains. This is a fundamental weakness of receiver-side spam protections: because the spammer will often send messages to different domains in an arbitrary order, the sheer multitude of possible target domains means that it will be very difficult to glean useful information from inter-spam arrival time as the probability of receiving spam messages sequentially from an individual botnet machine is very low. Intra-message response times, however, remain a possibility for useful inspection. Fundamentally, invariants that can be observed completely within individual delivery attempts will be the best features to use with this strategy, such as retry and failure modes, response to tarpitting, and network-level features like response to connection attempts on port 25.

Although the task of blacklisting hundreds of thousands of bots from limited vantage points is very difficult, domain specific dynamic blacklists and behavioral blacklisting offer insightful methods of leveraging network invariants within individual spammers' botnets to improve the coverage of traditional blacklists.

2.1.3 Sender Authentication

Authentication methods for anti-spam systems use a substrate of reliable sender identification in order to create whitelists or reputations for each sender or sending domain. As the usage model of SMTP (Simple Mail Transfer Protocol) allows for the relaying of mail on behalf of third parties, the standard does not require any association between the nominal sender of an email message (traditionally, the name given with the "From" command during the SMTP session) and the real sender (the entity who authored and initially submitted the message for delivery) [60]. While SMTP was extended to provide authentication within the protocol, this capability is mainly intended to restrict access to an organization's mail relay (the "outgoing" mail server which accepts messages from the organization's members and relays them to the "incoming" mail server of the recipient) [56].

Sender authentication at the granularity of individual senders [8] and organizations [45, 85] has been implemented, allowing recipients to verify the link between nominal sender and real sender. Authentication only provides a substrate upon which a reputation system can be implemented; policies must still be built upon this mechanism to determine which senders are spammers and which are delivering desired messages.

Traditionally, authorized sender whitelists were maintained manually. Much like other early anti-spam systems, the effectiveness of this mechanism has not scaled with the popularity of email. A modern inter-domain authorization must provide an automated, secure side channel for communication of authorization information; as this effort is attempting to associate the true email authors with their nominal senders, the modern systems outlined below have made the design decision to communicate authorization primitives through the DNS. Although this solves the problem of forged sender addresses, a reputation system for each sending organization must be built at each domain. The intuition behind this strategy is that if a large enough quorum of legitimate senders maintain a good reputation, poorly behaved senders can be easily filtered out while new good senders can still be introduced without difficulty. As initial deployment cannot expect instant worldwide adoption, the system outlined here uses strong authentication and reputation tracking as input to a comprehensive spam filtering mechanism.

Sender Reputation in a Large Webmail Service

Taylor illustrates the design decisions made when implementing a sender reputation system for *Gmail*, Google's free email service in [77]. This research highlights the bootstrapping and analysis of the use of sender authentication in production email environments. Gmail implements DomainKeys and the Sender Policy Framework (SPF) as both a sender and a receiver, using the results as input to their reputation mechanism. The author notes that nearly 75% of all legitimate mail is authenticated by at least one of Domain Keys or SPF.

Each of these authentication technologies allows a domain to verify that the nominal sender is the true sender of a given message, although each has its own weaknesses. SPF publishes a list of IP addresses or domain names under the sending domain's TXT record. While this provides a cryptography-free guarantee that a mail message was sent by an entity allowed to do so by the domain owner, if the message is forwarded before reaching the verifier (as is allowed in the SMTP protocol), the SPF check will fail. DomainKeys, conversely, allows the domain owner to publish a public key into the DNS that is used to sign outgoing messages rather than authenticate the sender himself. While a message body signed in this way (both the body and the headers of a message can be signed individually) will remain valid regardless of the number of forwarding hops, any modification by third parties during forwarding will break the signature. Thus, SPF and DomainKeys provide complementary services for email senders.

Empowered by a strong authentication for sending domains, Gmail builds the reputation for these domains with information both from user feedback within the mail interface itself and from statistical spam filters. The reputation for a given email message begins as the historical reputation for a given domain (calculated every few days), and is then corrected with any recent "this is spam" or "this is not spam" feedback for this domain. The reputation value is an integer between 0 and 100 which is a rough analog for the percent chance that an email from this domain is a legitimate message. Reputation values are acted upon for incoming messages using two threshold values: a whitelist threshold, above which all mails are delivered directly to the user's inbox; and a blacklist threshold, below which all mails are delivered directly to the user's spam folder. For messages between these thresholds, Gmail runs its statistical spam filters and makes a

final decision informed by the statistical filter and the reputation score for the sender.

Although the user feedback mechanism provides an excellent human-powered spam filter, it must still be tuned in order to provide the best information. User feedback which effects the reputation of any given sender is limited to one response per hour so that individual users cannot skew a domain's reputation. This mitigates, but does not prevent a spammer from marking his own messages as nonspam. The paper addresses this with clever reasoning, concluding it to be a non issue for the reputation system itself: if spammers are creating accounts on Gmail in order to whitelist their own spam, they might as well be using these accounts directly to send spam and are not explicitly a problem for the reputation system but rather for the abuse department.

2.1.4 Alternative Methods

Beyond the major techniques of classification, blacklisting, and reputation management, other methods have been suggested.

Proof of Work

So-called "proof of work" methods address the disconnect in resource allocation between senders and receivers in order to attest that the effort on the part of the sender is commensurate with the expected effort related to receiving and reading an email. Dwork and Naor suggested this strategy in the paper "Pricing via Processing" [13]. Proof of work methods require a pricing function for which computation of the output is a moderately difficult task, but verifying its correctness is easy. Any hard to solve but easy to verify function can be used; the popular "Hashcash" implementation requires the sender to find partial hash collisions for a given string, which can be best solved via brute force [5]. Proof of work has been generalized to use quotas alotted by a central authority, as well as shown in an economic sense to improve the overall utility of email to both senders and receivers due to the signaling process inherent in expending currency to send messages [48]. In addition to proof of work via processing or tokens alotted by a central authority, the same method can be abstracted into using any store of value as a pricing signal, including monetary "postage" for sending email [14].

While "proof of work" methods are a very promising theory, Laurie and Clayton

provide both theoretical and empirical evidence that such schemes have some important shortcomings [43]. Firstly, as spammers are using botnets to send most modern spam, the cost of the proof is not being directly borne by the spammer; compromised home machines typically have far more spare processing power than spare bandwidth. Secondly, empirical evidence presented in the paper shows that although 93.5% of all machines sent less than 75 emails per day (the global average), the top legitimate senders sent very many messages and more importantly did so in short bursts. While these senders would send messages more slowly or pay more to send messages in a proof of work system, spammers' messages are typically not as time sensitive and thus they will not experience as much quality of service degradation as the legitimate senders.

Address Management

Address management techniques build upon the fact that spam cannot easily be sent to an email address if the address itself is not available to the spammer. The Channels system exploits this fact by allowing an email user to create several unique email addresses, each of which can have its own policy regarding validity dates, authorized senders, and filtering strength [22]. While fundamental changes to the SMTP protocol are not necessary to implement channelized email addresses, as with some authentication schemes, multiple recipients and mailing lists pose problems as they leak possibly private email channels to third parties. The channel system uses a *personal channel agent* (PCA) for each user that maintains a database of active personal channels and manages the creation and publishing of new channels when necessary. While the email infrastructure does not require direct modification to support this scheme, the mail user agent must be able to communicate with the PCA in order to address email messages correctly. While the channel identifiers themselves are cryptographically secure against brute force guessing, the danger of address book eavesdropping via malware or other means on third parties presents the same risk as it does currently, although the Channels system does provide a protocol for automated channel switching to easily recover from the channel compromise. Other schemes provide similar services via a third party remailer server [18] which eliminates many of the multi-recipient problems, or via metadata in the address itself which encodes the policies associated with the address, e.g., which domain

is allowed to use the address and during which time period [27].

Challenge-response is a method closely related to addresses management techniques [53]. Challenge-response systems use custom email addresses in a similar manner to address management systems. However, instead of providing a complete anti-spam solution for individual users, challenge-response systems simply verify that the sender's email address was not forged by sending an automated reply asking the sender to verify that the communication is desired. A variant of this callback method is used an authentication scheme in the Occam protocol, where the callback mechanism is implemented as an independent, automated communication [15].

The fundamental information asymmetry of unsolicited communication and the unauthenticated, global nature of email assure us that undesired, unsolicited email will continue to exist for the forseeable future. Most current defenses are concerned primarily with filtering or blocking the messages themselves, but each of these messages has a sender with a goal in mind. But taking a step back and asking *why send spam* offers us key insights: a large proportion of modern spam messages advertise various goods for sale. Thus, spam can be understood not only as a technical problem, but as an economic one: spammers send their messages, presumably, because people will buy what they are selling. Understanding this *economic* approach to spam—how they make money, how much they stand to make—is key to finding new methods to decrease their bottom line, and decreasing in turn the number of spam messages sent every day.

2.2 The Economic Approach to Anti-Spam

The technical approach to anti-spam treats unsolicited messages themselves as the problem to be solved, and the goal is to effectively filter them. While this approach has been very successful for the organizations that deploy these filters, filters combat a symptom—receiving spam email—rather than the root cause: commercial spam is sent because it is an effective way to make money.

When viewed as part of a business operation, commercial spam shares many properties with direct marketing, a technique that dates back to the 19th century distribution of the first mail-order catalogs. What makes direct marketing so appealing is

that one can directly measure its return on investment. For example, the Direct Mail Association reports that direct mail sales campaigns produce a response rate of 2.15 percent on average [12]. Meanwhile, rough estimates of direct mail *cost per mille* (CPM) – the cost to address, produce and deliver materials to a thousand targets – range between $250 and $1000. Thus, following these estimates it might cost $250,000 to send out a million solicitations, which might then produce 21,500 responses. The cost of developing these prospects (roughly $12 each) can be directly computed and, assuming each prospect completes a sale of an average value, one can balance this revenue directly against the marketing costs to determine the profitability of the campaign. As long as the product of the conversion rate and the marginal profit per sale exceeds the marginal delivery cost, the campaign is profitable.

Given this underlying value proposition, it is not at all surprising that bulk direct email marketing emerged very quickly after email itself. The marginal cost to send an email is tiny and, thus, an email-based campaign can be profitable even when the conversion rate is negligible. Unfortunately, a perverse byproduct of this dynamic is that sending as much spam as possible is likely to maximize profit.

The resulting social nuisance begat a vibrant anti-spam community, eventually producing a multi-billion dollar industry focused on the same problem. However, with each anti-spam innovation spammers adapted in kind and, while the resulting co-evolution has not significantly changed the spam problem, it has changed how spam is purveyed. For example, the advent of real-time IP blacklisting deployed in Mail Transfer Agents (MTAs) forced spammers to relay their messages through "untainted" third-party hosts — driving the creation of modern large-scale botnets. Similarly, content-based anti-spam filters in turn forced spammers to create sophisticated polymorphism engines, modifying each spam message to be distinct. As well, it forced them to send even more spam. Thus, it has been estimated that over 120 billion spam messages are now sent each day [29].

However, while spam has long been understood to be an economic problem, it is only recently that there has been significant effort in modeling spam economics and understanding the value proposition from the spammer's point of view. Rarely do spammers talk about financial aspects of their activities themselves, though such accounts do exist [36, 83]. Judge et al. describe a prototypical model of spam profitability,

including both the basic value proposition as well as the impact of anti-spam filtering and law enforcement. They speculate that response rates as low as 0.000001 are sufficient to maintain profitability [30]. Khong [35] likewise employs an economic cost model of spam, comparing the success of several anti-spam strategies. Goodman and Rounthwaite construct a more complex model, aimed at deriving the cost factors for sending spam, and conclude depressingly that the optimal strategy for sending spam is to send as fast as possible [20]. Serjantov and Clayton explore these issues from the standpoint of an ISP and try to understand how to place appropriate incentives around the use of anti-spam blacklists [71].

Some data-based estimates of spammers do exist, but for the specialty type known as "stock spam" [7, 17, 23]. Stock spam refers to the practice of sending positive "touts" for a low-volume stock to manipulate its price and thereby profit on an existing position in the stock. What distinguishes stock spam is that it is monetized through price manipulation and not via a sale. Consequently, it is not necessary to measure the conversion rate to understand profitability. Instead, profitability can be inferred by correlating stock spam message volume with changes in the trading volume and price for the associated stocks.

While investigating stock spam has given researchers a data point regarding the profits of spammers, it is the modern commercial spam trade that accounts for the lion's share of modern unsolicited bulk email. Understanding how these modern spam campaigns work is necessary to begin investigating their success rates.

2.3 How spam-advertised sites work

Today, spam of all kinds represents an outsourced marketing operation in service to an underlying sales activity. At the core are "affiliate programs" that provide retail content (e.g., storefront templates and site code) as well as back-end services (e.g., payment processing, fulfillment and customer support) to a set of client affiliates. Affiliates in turn are paid on a commission basis (typically 30–50% in the pharmaceutical market) for each sale they bring in via whatever advertising vector they are able to harness effectively. This dynamic is well described in Samosseiko's "Partnerka" paper [69] and also in our

recent work studying the spam value chain [46].

Thus, while an affiliate has a responsibility to attract customers and host their shopping experience (which includes maintaining the contents of their "shopping cart"), once a customer decides to "check out" the affiliate hands the process over to the operators of the affiliate program.[1] Consequently, we would expect to find the order processing service shared across *all* affiliates of a particular program, regardless of the means used to attract customers. Indeed, as discussed below, our measurements of purchases from different members of the same affiliate confirm that the order numbers associated with the purchases come from a common pool. This finding is critical for our study because it means that side-effects in the order processing phase reflect the actions of *all sales activity* for an entire program, rather than just the sales of a single member.

On the back end, order processing consists of several steps: authorization, settlement, fulfillment, and customer service. Authorization is the process by which the merchant confirms, through the appropriate payment card association (e.g., Visa, MasterCard, American Express, Japan Credit Bureau, etc.), that the customer has sufficient funds. For the most common payment cards (Visa/MC), this process consists of contacting the customer's issuing bank, ensuring that the card is valid and the customer possesses sufficient funds, and placing a lien on the current credit balance. Once the good or service is ready for delivery, the merchant can then execute a settlement transaction that actualizes this lien, transferring money to the merchant's bank. Finally, fulfillment comprises packaging and delivery (e.g., shipping drugs directly from a foreign supplier or providing a Web site and password for downloading software). For our study, however, the key leverage lies in *customer service*. To support customer service, payment sites generate individual order numbers to share with the customer.

With this understanding, we can formulate hypotheses and methodologies for evaluating the effectiveness of each of these structural elements; the next section outlines why it is important to uncover these figures and how they can be used to create better defenses.

[1] This transfer typically takes the form of a redirection to a payment gateway site (with the affiliate's identity encoded in the request), although some sites also support a proxy mode so the customer can appear to remain at the same Web site.

2.4 Quantifying the Revenues of Cybercriminals

The security community is at once awash in the technical detail of new threats—the precise nature of a new vulnerability or the systematic analysis of a new botnet's command and control protocol—yet somewhat deficient in analyzing the economic processes that underlie these activities. In fairness, it is difficult to produce such analyses; there are innate operational complexities in acquiring such economic data and inherent uncertainties when reasoning about underground activities whose true scope is rarely visible directly.

However, absent a rigorous treatment, the resulting information vacuum is all too easily filled with opinion, which in turn can morph into "fact" over time. Though pervasive, this problem seemingly reached its zenith in the 2005 claim by US Treasury Department consultant Valerie McNiven that cybercrime revenue exceeded that of the drug trade (over $100 billion at the time) [34]. This claim was frequently repeated by members of the security industry, growing in size each year, ultimately reaching its peak in 2009 with written Congressional testimony by AT&T's chief security officer stating that cybercrime reaped "more than $1 trillion annually in illicit profits" [70]—a figure well in excess of the entire software industry and almost twice the GDP of Germany. Nay-sayers are similarly limited in their empirical evidence. Perhaps best known in this group are Herley and Florencio, who argue that a variety of cybercrimes are generally unprofitable. However, lacking empirical data, they are forced to use an economic meta-analysis to make their case [24, 25, 26].

2.4.1 Estimating spam revenue and demand

Public estimates of the revenue of spammers and demand for their products can vary widely. In 2005, one consultancy estimated that Russian spammers earned roughly US$2–3M per year [50]. However, in a 2008 interview, one IBM representative claimed that a single spamming botnet was earning close to $2M *per day* [1]. Our previous work studied the same botnet empirically, leading to an estimate of daily revenue of up to $9,500, extrapolating to $3.5M *per year* [32]. Most recently, a report by the Russian Association of Electronic Communication (RAEC) estimated that Russian spammers

earned 3.7 billion rubles (roughly $125 million) in 2009 [38].

The demand side of this equation is even less well understood, relying almost entirely on opt-in phone or email polls. In 2004, the Business Software Alliance sponsored a Forrester Research poll to examine this question, finding that out of 6,000 respondents (spread evenly across the US, Canada, Germany, France, the UK and Brazil) 27% had purchased spam-advertised software and 13% had purchased spam-advertised pharmaceuticals [16]. If such data were taken at face value, the US market size for spam-advertised pharmaceuticals would exceed 30 million customers. Similar studies, one by Marshal in 2008 and the other sponsored by the Messaging Anti-Abuse Working Group (MAAWG) in 2009, estimate that 29% and 12%, respectively, of Internet users had purchased goods or services advertised in spam email [28, 52].

Understanding the effectiveness of spam and the size of its market, both on the supply and demand sides, is crucial to informing correct response to the spam problem. Without an "evidence basis", policy and investment decisions are easily distorted along influence lines, either over-reacting to small problems or under-appreciating the scope of grave ones. With this goal in mind, the next chapter presents both our methodology for and results from measuring the conversion rate of a botnet-based spam campaign, providing insight into the immense scale of modern spam campaigns and beginning to uncover details of the demand that supports this enterprise.

Chapter 2, in part, is a reprint of the material as it appears in Proceedings of ACM CCS 2008. Kanich, Chris; Kreibich, Christian; Levchenko, Kirill; Enright, Brandon; Voelker, Geoffrey M.; Paxson, Vern; Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 2, in part, is a reprint of the material as it appears in Proceedings of USENIX Security 2010. Kanich, Chris; Weaver, Nicholas; McCoy, Damon; Halvorson, Tristan; Kreibich, Christian; Levchenko, Kirill; Paxson, Vern; Voelker, Geoffrey M., Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 3

The Economics of a Modern Spam Campaign

Spam-based marketing is a curious beast. We all receive the advertisements — "Excellent hardness is easy!" — but few of us have encountered a person who admits to following through on this offer and making a purchase. And yet, the relentlessness by which such spam continually clogs Internet inboxes, despite years of energetic deployment of anti-spam technology, provides undeniable testament that spammers find their campaigns profitable. Someone is clearly buying. But how many, how often, and how much?

Unraveling such questions is *essential* for understanding the economic support for spam and hence where any structural weaknesses may lie. Unfortunately, spammers do not file quarterly financial reports, and the underground nature of their activities makes third-party data gathering a challenge at best. Absent an empirical foundation, defenders are often left to speculate as to how successful spam campaigns are and to what degree they are profitable. For example, IBM's Joshua Corman was widely quoted as claiming that spam sent by the Storm worm alone was generating "millions and millions of dollars every day" [1]. While this claim could in fact be true, we are unaware of any public data or methodology capable of confirming or refuting it.

The key problem is our limited visibility into the three basic parameters of the spam value proposition: the cost to send spam, offset by the "conversion rate" (probability that an email sent will ultimately yield a "sale"), and the marginal profit per sale. The

first and last of these are self-contained and can at least be estimated based on the costs charged by third-party spam senders and through the pricing and gross margins offered by various Internet marketing "affiliate programs".[1] However, the conversion rate depends fundamentally on group actions — on what hundreds of millions of Internet users do when confronted with a new piece of spam — and is much harder to obtain. While a range of anecdotal numbers exist, we are unaware of any well-documented measurement of the spam conversion rate.[2]

In part, this problem is methodological. There are no apparent methods for indirectly measuring spam conversion. Thus, the only obvious way to extract this data is to build an e-commerce site, market it via spam, and then record the number of sales. Moreover, to capture the spammer's experience with full fidelity, such a study must also mimic their use of illicit botnets for distributing email and proxying user responses. Through *botnet infiltration*, we achieved this goal, and this chapter presents our methodology and resulting analysis.

The remainder of this chapter is structured as follows. Section 3.1 describes the Storm botnet, and Section 3.2 describes our infrastructure for transparently interposing on communication between captive malware and the Internet. Section 3.3 describes how we use this infrastructure to infiltrate and study the operation of the Storm botnet. Section 3.4 describes our spam filtering and conversion results, Section 3.5 analyzes the effects of blacklisting on spam delivery, and Section 3.6 analyzes the possible influences on spam responses. We summarise the findings of this research effort in Section 3.7.

3.1 The Storm Botnet

The measurements in this project are carried out using the Storm botnet and its spamming agents. A full technical overview of the operation of the Storm botnet is presented in [75]; here we review key mechanisms in Storm's communication protocols and organizational hierarchy that are relevant to our botnet infiltration technique.

[1] Our cursory investigations suggest that commissions on pharmaceutical affiliate programs tend to hover around 40-50%, while the *retail* cost for spam delivery has been estimated at under $80 per million [84].

[2] The best known among these anecdotal figures comes from the Wall Street Journal's 2003 investigation of Howard Carmack (a.k.a the "Buffalo Spammer"), revealing that he obtained a 0.00036 conversion rate on ten million messages marketing an herbal stimulant [3].

Storm is a peer-to-peer botnet that propagates via spam (usually by directing recipients to download an executable from a Web site). Storm communicates using two separate protocols: the first is an encrypted version of the UDP-based Overnet protocol (in turn based on the Kademlia DHT [54]) and is used primarily as a directory service to find other nodes. As well, Storm uses a custom TCP-based protocol for managing command and control — the directions informing each bot what actions it should take. We describe each of these below.

3.1.1 Overnet protocol

There are four basic messages to facilitate the basic functioning of Overnet: *Connect*, *Search*, *Publicize*, and *Publish*. During the bootstrap phase, a Storm node only has the initial list of peers that it was shipped with. To gather more peers Storm chooses an address for itself pseudo-randomly from the 128-bit Overnet address space and proceeds to *Connect* to all the peers in its bootstrap list. Each available peer contacted returns a list of up to 20 peers. Storm does this for a few rounds until it has gathered enough peers to be adequately connected in Overnet. Once a new node has learned about enough peers it switches to *Publicizing* its presence to nearby peers and periodically searching for its own address to stay connected and learn about new close-by peers to keep up with churn.

Overnet also provides two messages for storing and finding content in the network: *Publish* and *Search* which export a standard DHT (key,value) pair interface. However, Storm uses this interface in an unusual way. In particular, the keys encode a dynamically changing rendezvous code that allow Storm nodes to find each other on demand.

A Storm node generates and uses three rendezvous keys simultaneously: one based on the current date, one based on the previous date, and one based on the next date. To determine the correct date, Storm first sets the system clock using NTP.

In particular, each key is based on a combination of the time (with 24-hour resolution) mixed with a random integer between 0 and 31. Thus there are 32 unique Storm keys in use per day but a single Storm bot will only use 1 of the 32. Because keys are based on time, Storm uses NTP to sync a bot's clock and attempts to normalize the time zone. Even so, to make sure bots around the world can stay in sync, Storm uses 3 days of keys at once, the previous, current, and next day.

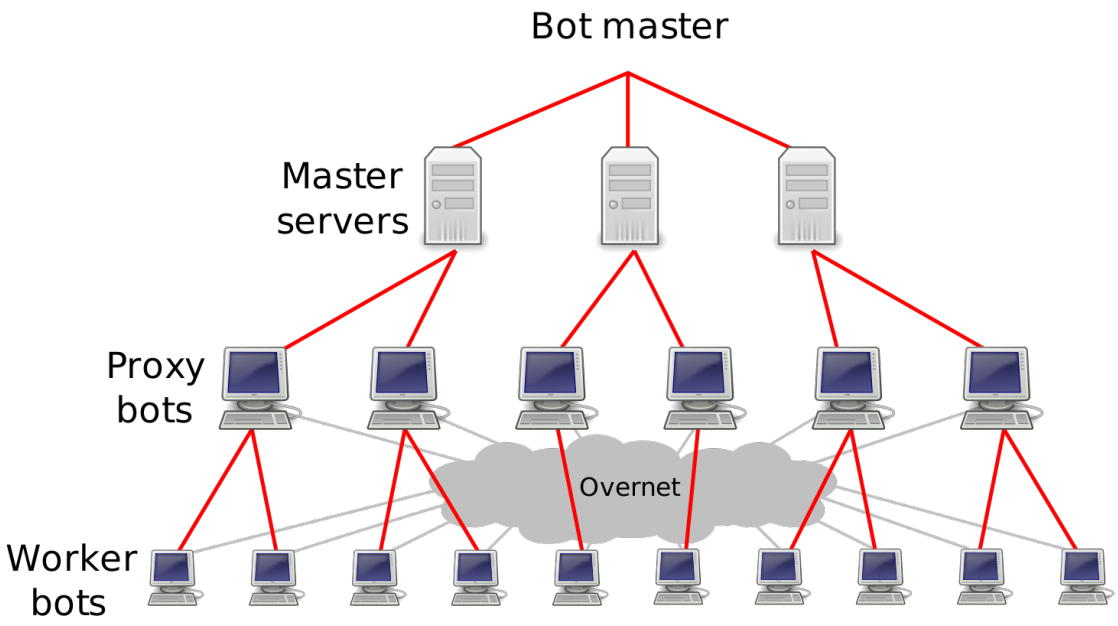


Figure 3.1: The Storm botnet hierarchy.

In turn, these keys are used to rendezvous with Storm nodes that implement the command and control (C&C) channel. A Storm node that wishes to offer the C&C service will use the time-based hashing algorithm to generate a key and encode its own IP address and TCP port into the value. It will then search for the appropriate peers close to the key and publish its (key, value) pair to them. A peer wishing to locate a C&C channel can generate a time-based key and search for previously published values to decode and connect to the TCP network.

3.1.2 Storm hierarchy

There are three primary classes of Storm nodes involved in sending spam (shown in Figure 3.1). Worker bots make requests for work and, upon receiving orders, send spam as requested. Proxy bots act as conduits between workers and master servers. Finally, the master servers provide commands to the workers and receive their status reports. In our experience there are a very small number of master servers (typically hosted at so-called "bullet-proof" hosting centers) and these are likely managed by the botmaster directly.

However, the distinction between worker and proxy is one that is determined automatically. When Storm first infects a host it tests if it can be reached externally. If so,

then it is eligible to become a proxy. If not, then it becomes a worker.

3.1.3 Spam engine

Having decided to become a worker, a new bot first checks whether it can reach the SMTP server of a popular Web-based mail provider on TCP port 25. If this check fails the worker will remain active but not participate in spamming campaigns.[3]

Figure 3.5 outlines the broad steps for launching spam campaigns when the port check is successful. The worker finds a proxy (using the time-varying protocol described earlier) and then sends an update request (via the proxy) to an associated master server (Step 1), which will respond with a spam workload task (Step 2). A spam workload consists of three components: one or more spam templates, a delivery list of email addresses, and a set of named "dictionaries". Spam templates are written in a custom macro language for generating polymorphic messages [40]. The macros insert elements from the dictionaries (e.g., target email addresses, message subject lines), random identifiers (e.g., SMTP message identifiers, IP addresses), the date and time, etc., into message fields and text. Generated messages appear as if they originate from a valid MTA, and use polymorphic content for evading spam filters.

Upon receiving a spam workload, a worker bot generates a unique message for each of the addresses on the delivery list and attempts to send the message to the MX of the recipient via SMTP (Step 3). When the worker bot has exhausted its delivery list, it requests two additional spam workloads and executes them. It then sends a delivery report back to its proxy (Step 4). The report includes a result code for each attempted delivery. If an attempt was successful, it includes the full email address of the recipient; otherwise, it reports an error code corresponding to the failure. The proxy, in turn, relays these status reports back to the associated master server.

To summarize, Storm uses a three-level self-organizing hierarchy comprised of worker bots, proxy bots and master servers. Command and control is "pull-based", driven by requests from individual worker bots. These requests are sent to proxies who, in turn, automatically relay these requests to master servers and similarly forward any attendant

[3] Such bots are still "useful" for other tasks such as mounting coordinated DDoS attacks that Storm perpetrates from time to time.

responses back to the workers.

3.2 Botnet Infiltration Infrastructure

To infiltrate the Storm Botnet, we extended the GQ malware execution farm [42], introducing new capabilities allowing the monitoring, containment, and modification of Internet communication flows between infected virtual machines and the commodity Internet.

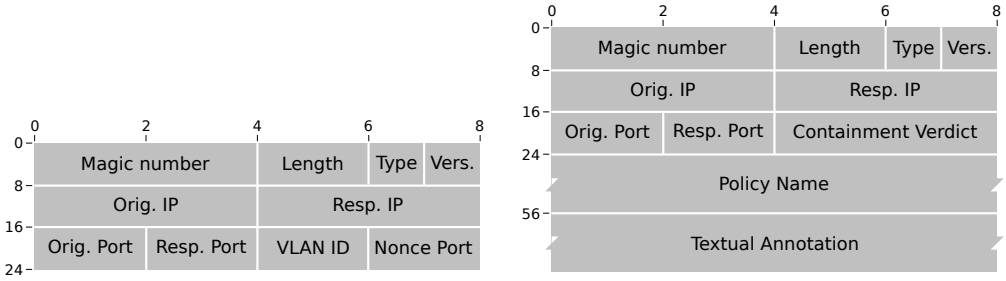
As we must allow the Storm malware to communicate with hosts on the commodity Internet, we must guarantee that the botnet does not harm any other hosts during our experiments. We present GQ's containment capabilities which enable our infiltration of the Storm botnet in this section.

3.2.1 Containment server implementation

We wrote the containment server in Python, using a pre-forked, multi-threaded service model. The server logic comprises roughly 2,200 lines, with 600 lines for the event-trigger logic. The containment policies, including content rewriters, add up to 1,000 lines.

Shimming protocol. To couple the gateway's packet router to the containment server, we need a way to map/unmap arbitrary flows to/from the single address and port of the containment server. We achieve this mapping using a *shimming protocol* conceptually similar to SOCKS [37]: upon redirection to the containment server, the gateway injects into the flow a *containment request* shim message with meta-information. Figure 3.2 summarizes the message structure.

The containment server expects this meta-information and uses it to assign a containment policy to the flow. The containment server similarly conveys the containment verdict back to gateway using a *containment response* shim, which the packet router strips from the flow before relaying subsequent content back to the endpoint. For TCP, the shim is sent as a separate TCP packet injected into the sequence space, so adding and removing shim packets requires bumping and unbumping of sequence and acknowledgement



(a) Request Shim (b) Response Shim

Figure 3.2: The shim protocol message structure.

numbers; for UDP it requires padding the datagrams with the respective shims.[4] The request shim occupies 24 bytes and begins with a preamble of 8 bytes containing a magic number (4 bytes), the message length (2 bytes), a message type indicator (1 byte), and a shim protocol version number (1 byte), followed by the original flow's endpoint four-tuple ($2 \cdot 4$ bytes plus $2 \cdot 2$ bytes), the VLAN ID of the sending/receiving inmate (2 bytes) and a nonce port (2 bytes) on which the gateway will expect a possible subsequent outbound connection from the containment server, in case the latter needs to rewrite the flow continuously. The response shim can vary in length and requires at least 56 bytes, consisting of a similar preamble (8 bytes), the resulting endpoint four-tuple (12 bytes), the containment verdict (`FORWARD`, `LIMIT`, `DROP`, `REDIRECT`, `REFLECT`, or `REWRITE`, possibly in combination when feasible) expressed as a numeric opcode (4 bytes), a name tag for the resulting containment policy (32 bytes), and an optional annotation string to clarify the context in which the containment server decided the containment verdict.

We show an example of this containment procedure in Figure 3.3. An inmate initiates the TCP handshake for an upcoming HTTP request (❶), which the gateway redirects to the containment server's fixed address and port, synthesizing a full TCP handshake. Upon completion, the gateway injects into the TCP stream the containment request shim (❷). The containment server in turn sends a containment response shim (❸) including the containment verdict for the flow, in this case a `REWRITE`. To serve as a transparent proxy rewriting the flow content, it also establishes a second TCP connection to the target via the gateway and the nonce port received as part of the containment

[4] For large UDP datagrams, this can require fragmentation.

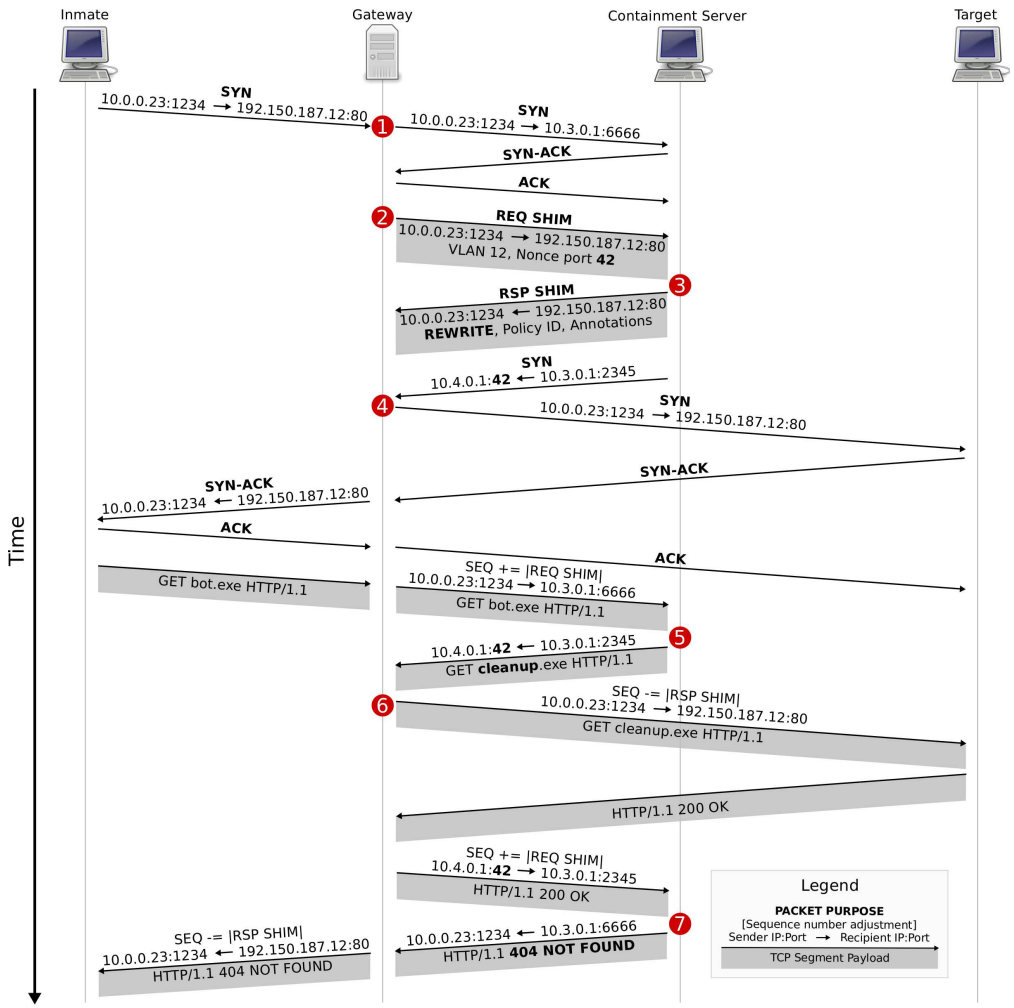


Figure 3.3: TCP packet flow through gateway and containment server in a `REWRITE` containment.

request shim. The gateway forwards this TCP `SYN` to the target and relays the handshake between target and inmate (❹). The inmate completes its connection establishment and sends the HTTP request, which the gateway relays on to the containment server as part of the same connection that it used to exchange containment information, bumping the sequence number accordingly (❺). The containment server rewrites the request as needed (here changing the requested resource to another one) and forwards it on to the target, via the gateway (❻). The target's response travels in the opposite direction and arrives at the containment server, which again rewrites it (here to create the illusion of a non-existing resource) and relays it back to the inmate (❼). (For brevity, we do not show the subsequent connection tear-downs in the figure.)

```
[VLAN 16-17]
Decider = Rustock
Infection = rustock.100921.*.exe

[VLAN 18-19]
Decider = Grum
Infection = grum.100818.*.exe

[VLAN 16-19]
Trigger = *:25/tcp / 30min < 1 -> revert

[Autoinfect]
Address = 10.9.8.7
Port = 6543

[BannerSmtpSink]
Address = 10.3.1.4
Port = 2526
```

Figure 3.4: Example of a containment server configuration file: GQ will infect the inmates on VLAN IDs 16 and 17 iteratively with binaries from the `rustock.100921.*.exe` batch, using the "Rustock" containment policy. It will apply similar policies to inmates with VLAN IDs 18 and 19, for Grum. On all four VLAN IDs a lifecycle trigger reverts inmates to a clean state whenever the number of flows to TCP port 25 in 30 minutes hits zero. The last two sections specify the location in the subfarm of an auto-infection server and of an SMTP sink, respectively.

Policy structure. We codify containment policies in Python classes, which the containment server instantiates by keying on VLAN ID ranges and applies on a per-flow basis. We base endpoint control upon the flow's four-tuple, and content control depends on the actual data sent in a given flow. Object-oriented implementation reuse and specialization lends itself well to the establishment of a hierarchy of containment policies. From a base class implementing a default-deny policy we derive classes for each endpoint control verdict, and from these specialize further, for example to a base class for spambots that reflects all outbound SMTP traffic. The containment server simply takes the name of the class implementing the applicable containment policy into the response shim (recall Figure 3.2b) in order to convey it to the gateway.

Configuration. The codified containment policies are customizable through a configuration file. This file serves four purposes. It specifies: (*i*) the initial assignment of a policy to a given inmate's traffic, (*ii*) the individual or set of malware binaries with which we would like to infect a given inmate over the course of its life-cycles,[5] (*iii*) activity triggers (e.g., revert and reinfect the inmate once the containment server has observed no outbound activity for 30 minutes) and (*iv*) IP addresses and port numbers of infrastructure services in a subfarm (e.g., where to find a particular SMTP sink or HTTP proxy). Figure 3.4 shows a sample configuration snippet.

3.3 Methodology

Our measurement approach is based on *botnet infiltration* — we combine our understanding of the Storm botnet outlined in Section 3.1 with the GQ malware farm introduced in Section 3.2 to insinuate ourselves into the botnet's "command and control" (C&C) network, passively observing the spam-related commands and data it distributes and, where appropriate, actively changing individual elements of these messages in transit. Storm's architecture lends itself particularly well to infiltration since the proxy bots, *by design*, interpose on the communications between individual worker bots and the master servers who direct them. Moreover, since Storm compromises hosts indiscriminately (normally using malware distributed via social engineering Web sites) it is straightforward to create a proxy bot on demand by infecting a globally reachable host under our control with the Storm malware.

Figure 3.5 also illustrates our basic measurement infrastructure. At the core, we instantiate eight unmodified Storm proxy bots within a controlled virtual machine environment hosted on VMWare ESX 3 servers. The network traffic for these bots is then routed through a centralized gateway, providing a means for blocking unanticipated behaviors (e.g., participation in DDoS attacks) and an interposition point for parsing C&C messages and "rewriting" them as they pass from proxies to workers. Most critically, by carefully rewriting the spam template and dictionary entries sent by master servers,

[5] As indicated earlier, we typically specify precisely which sample to infect an inmate with. However, GQ equally supports traditional honeypot constellations in which dynamic circumstances (such as a web drive-by) determine the nature of the infection.

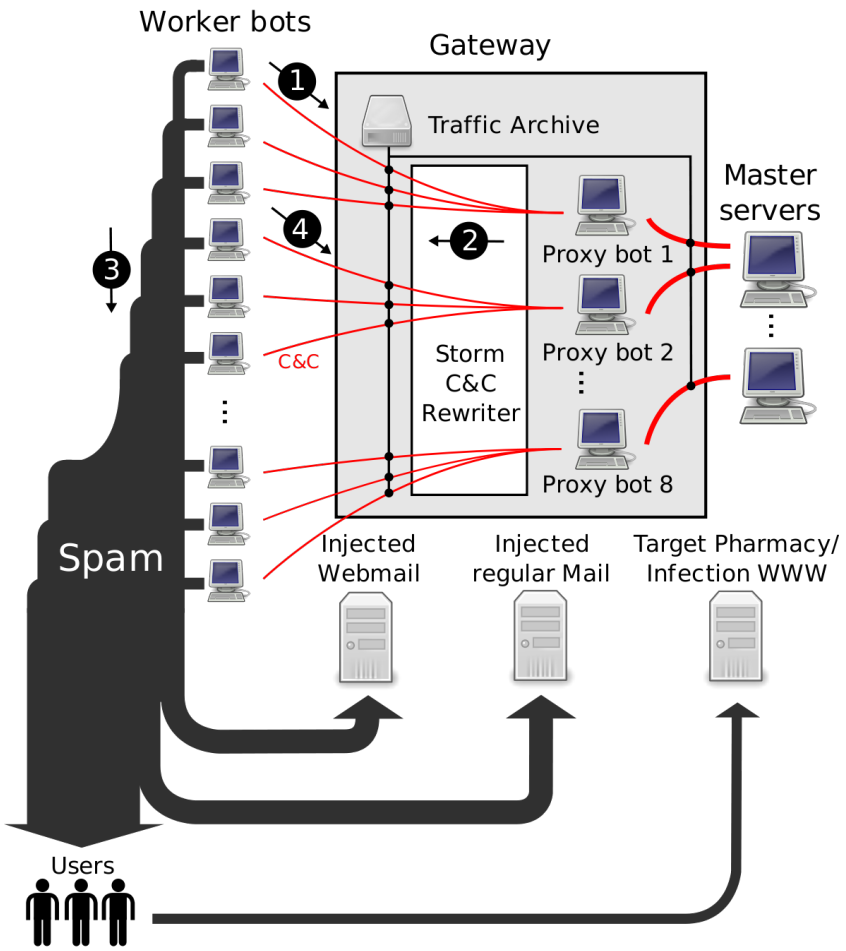


Figure 3.5: The Storm spam campaign dataflow (Section 3.1.3) and our measurement and rewriting infrastructure (Section 3.3). (1) Workers request spam tasks through proxies, (2) proxies forward spam workload responses from master servers, (3) workers send the spam and (4) return delivery reports. Our infrastructure infiltrates the C&C channels between workers and proxies.

we arrange for worker bots to replace the intended site links in their spam with URLs of our choosing. From this basic capability we synthesize experiments to measure the click-through and conversion rates for several large spam campaigns.

In the remainder of this section we provide a detailed description of our Storm C&C rewriting engine, discuss how we use this tool to obtain empirical estimates for spam delivery, click-through and conversion rates and describe the heuristics used for differentiating real user visits from those driven by automated crawlers, honey-clients, etc. With this context, we then review the ethical basis upon which these measurements

were conducted.

3.3.1 C&C protocol rewriting

Our runtime C&C protocol rewriter consists of two components. A custom Click-based network element redirects potential C&C traffic to a fixed IP address and port, where a user-space proxy server implemented in Python accepts incoming connections and impersonates the proxy bots. This server in turn forwards connections back into the Click element, which redirects the traffic to the intended proxy bot. To associate connections to the proxy server with those forwarded by the proxy server, the Click element injects a SOCKS-style destination header into the flows. The proxy server uses this header to forward a connection to a particular address and port, allowing the Click element to make the association. From that point on, traffic flows transparently through the proxy server where C&C traffic is parsed and rewritten as required. Rules for rewriting can be installed independently for templates, dictionaries, and email address target lists. The rewriter logs all C&C traffic between worker and our proxy bots, between the proxy bots and the master servers, and all rewriting actions on the traffic.

Since C&C traffic arrives on arbitrary ports, we designed the proxy server so that it initially handles any type of connection and falls back to passive pass-through for any non-C&C traffic. Since the proxy server needs to maintain a connection for each of the (many) workers, we use a preforked, multithreaded design. A pool of 30 processes allowed us to handle the full worker load for the eight Storm proxy bots at all times.

3.3.2 Measuring spam delivery

To evaluate the effect of spam filtering along the email delivery path to user inboxes, we established a collection of test email accounts and arranged to have Storm worker bots send spam to those accounts. We created multiple accounts at three popular free email providers (Gmail, Yahoo!, and Hotmail), accounts filtered through our department commercial spam filtering appliance (a Barracuda Spam Firewall Model 300 with slightly more permissive spam tagging than the default setting), and multiple SMTP “sinks” at distinct institutions that accept any message sent to them (these served

as “controls” to ensure that spam emails were being successfully delivered, absent any receiver-side spam filtering). When worker bots request spam workloads, our rewriter appends these email addresses to the end of each delivery list. When a worker bot reports success or failure back to the master servers, we remove any success reports for our email addresses to hide our modifications from the botmaster.

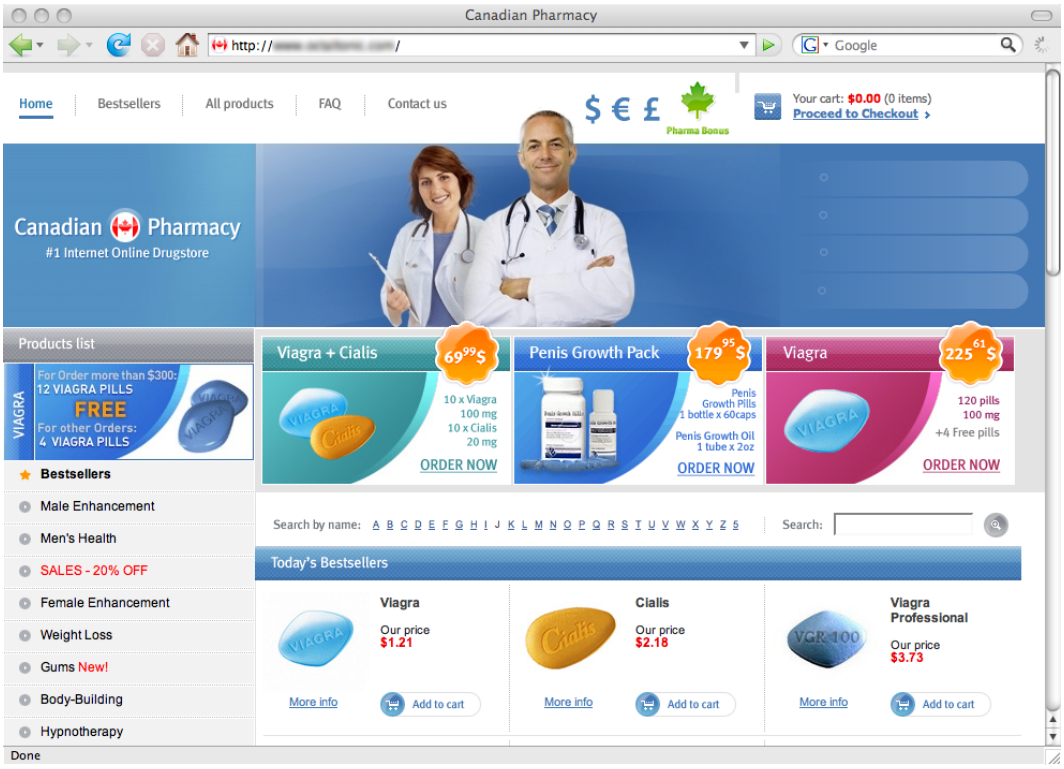
We periodically poll each email account (both inbox and “junk/spam” folders) for the messages that it received, and we log them with their timestamps. However, some of the messages we receive have nothing to do with our study and must be filtered out. These messages occur for a range of reasons, including spam generated by “dictionary bots” that exhaustively target potential email addresses, or because the addresses we use are unintentionally “leaked” (this can happen when a Storm worker bot connects to our proxy and then leaves before it has finished sending its spam; when it reconnects via a new proxy the delivery report to the master servers will include our addresses). To filter such email, we validate that each message includes both a subject line used by our selected campaigns and contains a link to one of the Web sites under our control.

3.3.3 Measuring click-through and conversion

To evaluate how often users who receive spam actually visit the sites advertised requires monitoring the advertised sites themselves. Since it is generally impractical to monitor sites not under our control, we have arranged to have a fraction of Storm’s spam advertise sites of our creation instead.

In particular, we have focused on two types of Storm spam campaigns, a self-propagation campaign designed to spread the Storm malware (typically under the guise of advertising an electronic postcard site) and the other advertising a pharmacy site. These are the two most popular Storm spam campaigns and represent over 40% of recent Storm activity [40].

For each of these campaigns, the Storm master servers distribute a specific “dictionary” that contains the set of target URLs to be inserted into spam emails as they are generated by worker bots. To divert user visits to our sites instead, the rewriter replaces any dictionaries that pass through our proxies with entries only containing URLs to our Web servers.



(a) Pharmaceutical site

(b) Postcard-themed self-propagation site

Figure 3.6: Screenshots of the Web sites operated to measure user click-through and conversion.

In general, we strive for verisimilitude with the actual Storm operation. Thus, we are careful to construct these URLs in the same manner as the real Storm sites (whether this is raw IP addresses, as used in the self-propagation campaigns, or the particular "nounnoun.com" naming schema used by the pharmacy campaign) to ensure the generated

spam is qualitatively indistinguishable from the "real thing". An important exception, unique to the pharmacy campaign, is an identifier we add to the end of each URL by modifying the associated spam template. This identifier allows us to unambiguously associate individual spam messages with subsequent accesses to the site. We did not add this identifier to the self-propagation campaigns since their URLs typically consist entirely of raw IP addresses. The addition of a text identifier suffix might thus appear out of place, reducing verisimilitude, and perhaps bias user click behavior.

Finally, we created two Web sites to mimic those used in the associated campaigns (screenshots of these sites are shown in Figure 3.6). The pharmaceutical site, primarily marketing "male-enhancement" drugs such as Viagra, is a nearly-precise replica of the site normally advertised by Storm down to using the same naming convention for the domains themselves. Our site mirrors the original site's user interface, the addition of products advertised for sale to a "shopping cart", and navigation up to, but not including, the input of personal and payment information (there are a range of complex regulatory, legal and ethical issues in accepting such information). Instead, when a user clicks on "Checkout" we return a 404 error message. We log all accesses to the site, allowing us to determine when a visitor attempts to make a purchase and what the content of their shopping cart is at the time. We assume that a purchase attempt is a conversion, which we speculate is a reasonable assumption, although our methodology does not allow us to validate that the user would have actually completed the purchase or that their credit card information would have been valid.

The self-propagation campaign is Storm's key mechanism for growth. The campaign entices users to download the Storm malware via deception; for example by telling them it is postcard software essential for viewing a message or joke sent to them by a friend. Unlike the pharmacy example, we were not able to mirror the graphical content of the postcard site, since it was itself stolen from a legitimate Internet postcard site. Instead, we created a close analog designed to mimic the overall look and feel. We also "defanged" our site by replacing its link to the Storm malware with that of a benign executable. If run, our executable is designed to performs a simple HTTP POST with a harmless payload ("`data=1`") to a server under our control, and then exit. As a rough timeout mechanism, the executable will not send the message if the system date is 2009 or later. Since the

postcard site we impersonated served three different executables under different names, we served three executables with different target filenames in the POST command as well. Again, all accesses to the site are logged and we are able to identify when our binary has been downloaded. Moreover, by correlating with the POST signal, we are able to determine if a particular download is ultimately executed on the visitor's machine (and hence is a conversion). Downloads and executions can differ because the user has second thoughts about allowing an execution or because the user's security software prevents it from executing (indeed, we observed that several anti-virus vendors developed signatures for our benign executable within a few days of our introducing it).

3.3.4 Separating users from crawlers

As with our email accounts, not all visits to our Web site are prospective conversions. There is a range of automated and semi-automated processes that visit our sites, ranging from pure Web crawlers, to "honeyclient" systems designed to gather intelligence on spam advertised sites, to security researchers trying to identify new malware.

To filter out such visits (which we generically call "crawlers") from intentful ones, we have developed a series of heuristics to identify crawlers and use this data to populate a global IP blacklist across all of our Web sites. We outline these heuristics below.

First, we consider all hosts that access the pharmacy site that do not use a URL containing the unique identifier discussed in Section 3.3.3 to be crawlers. Second, we blacklist hosts that access `robots.txt` (site-specific instructions meant only for Web crawlers) and hosts that make malformed requests (most often exploit attempts). Third, we blacklist all hosts that disable javascript and do not load embedded images. We assume that typical users do not browse under these conditions, whereas some large-scale anti-spam honeypots that follow embedded links in suspected spam exhibit this behavior to reduce load.

In addition to blacklisting based on the behavior of individual site visits, another common pattern we observed was the same IP address accessing the pharmacy site using several different unique identifiers, presumably as part of a spam defense or measurement mechanism. Consequently, we blacklist an IP address seen accessing the pharmacy site with more than one unique identifier with the same `User-Agent` field. This heuristic

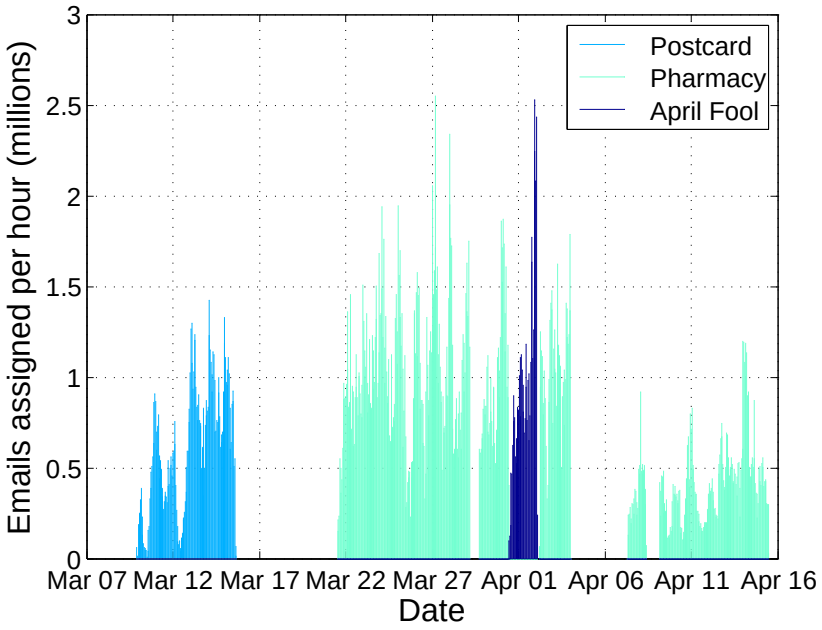


Figure 3.7: Number of email messages assigned per hour for each campaign.

does not filter users browsing behind larger Web proxy services, but does filter the homogeneous accesses seen from spam honeyclients. Similarly, we also blacklist any host that requests the downloaded executable from the postcard site ten or more times, under the assumption that such hosts are used by researchers or other observers interested in tracking updates to the Storm malware.

Finally, it has become common for anti-malware researchers to find new versions of the Storm malware by directly accessing the self-propagation dictionary entries. To detect such users we injected new IP addresses (never advertised in spam messages) into the self-propagation dictionary during a period of inactivity (i.e., when no self-propagation spam was being sent). Any visitors to these IP addresses could not have resulted from spam, and we therefore also added them to our crawler blacklist.

It is still possible that some of the accesses were via full-featured, low-volume honeyclients, but even if these exist we believe they are unlikely to significantly impact the data.

Table 3.1: Campaigns used in the experiment.

CAMPAIGN	DATES	WORKERS	EMAILS
Pharmacy	Mar 21 – Apr 15	31,348	347,590,389
Postcard	Mar 9 – Mar 15	17,639	83,665,479
April Fool	Mar 31 – Apr 2	3,678	38,651,124
		Total	469,906,992

3.3.5 Measurement ethics

We have been careful to design experiments that we believe are both consistent with current U.S. legal doctrine and are fundamentally ethical as well. While it is beyond the scope of this paper to fully describe the complex legal landscape in which active security measurements operate, we believe the ethical basis for our work is far easier to explain: *we strictly reduce harm*. First, our instrumented proxy bots do not create any new harm. That is, absent our involvement, the same set of users would receive the same set of spam emails sent by the same worker bots. Storm is a large self-organizing system and when a proxy fails its worker bots automatically switch to other idle proxies (indeed, when our proxies fail we see workers quickly switch away). Second, our proxies are passive actors and do not themselves engage in any behavior that is intrinsically objectionable; they do not send spam email, they do not compromise hosts, nor do they even contact worker bots asynchronously. Indeed, their only function is to provide a conduit between worker bots making requests and master servers providing responses. Finally, where we do modify C&C messages in transit, these actions themselves strictly reduce harm. Users who click on spam altered by these changes will be directed to one of our innocuous doppelganger Web sites. Unlike the sites *normally* advertised by Storm, our sites do not infect users with malware and do not collect user credit card information. Thus, no user should receive more spam due to our involvement, but some users will receive spam that is less dangerous that it would otherwise be.

At the same time, our infrastructure and modifications only strictly reduce harm to actual users. The proxy bots we operate do not send spam — the proxies are relays, and only the worker bots actually send spam. The existence of our proxy bots does not enable any additional spam to be sent — the workers connected to our proxies would have simply connected to different proxies otherwise. Further, we do not cause spam to

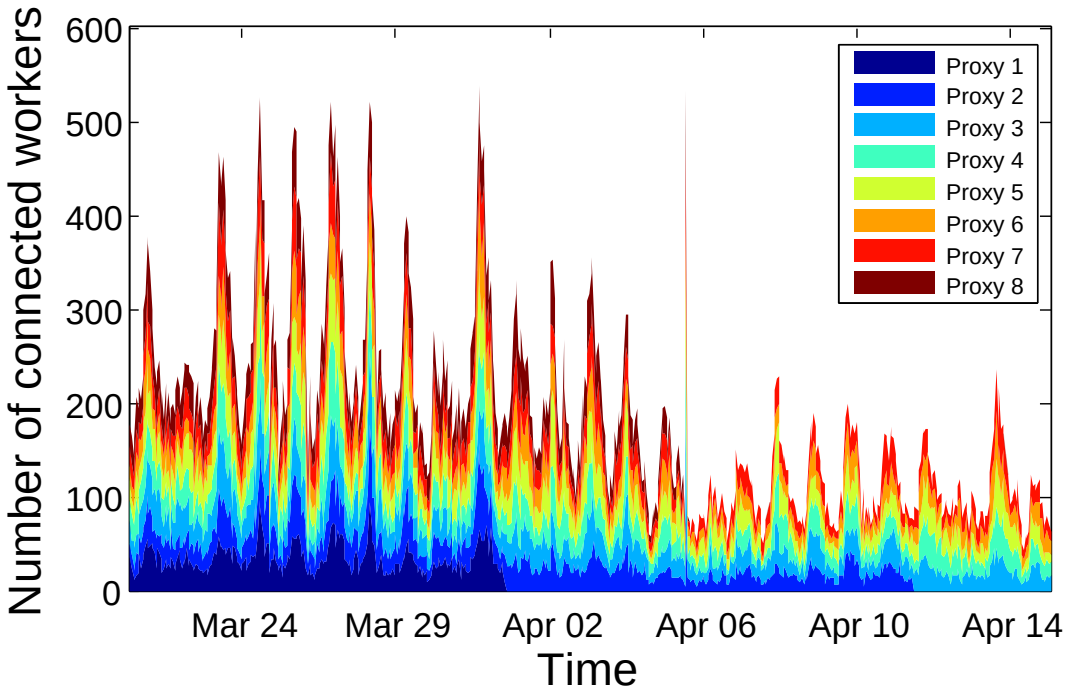


Figure 3.8: Timeline of proxy bot workload.

be sent to users who would not have otherwise been targeted, only to additional accounts under our personal control. Most importantly, we render the spam sent by the worker bots harmless by changing the URLs included in messages to refer to innocuous sites rather than sites that contain malware or collect personally identifying information.

3.4 Experimental results

We now present the overall results of our rewriting experiment. We first describe the spam workload observed by our C&C rewriting proxy. We then characterize the effects of filtering on the spam workload along the delivery path from worker bots to user inboxes, as well as the number of users who browse the advertised Web sites and act on the content there.

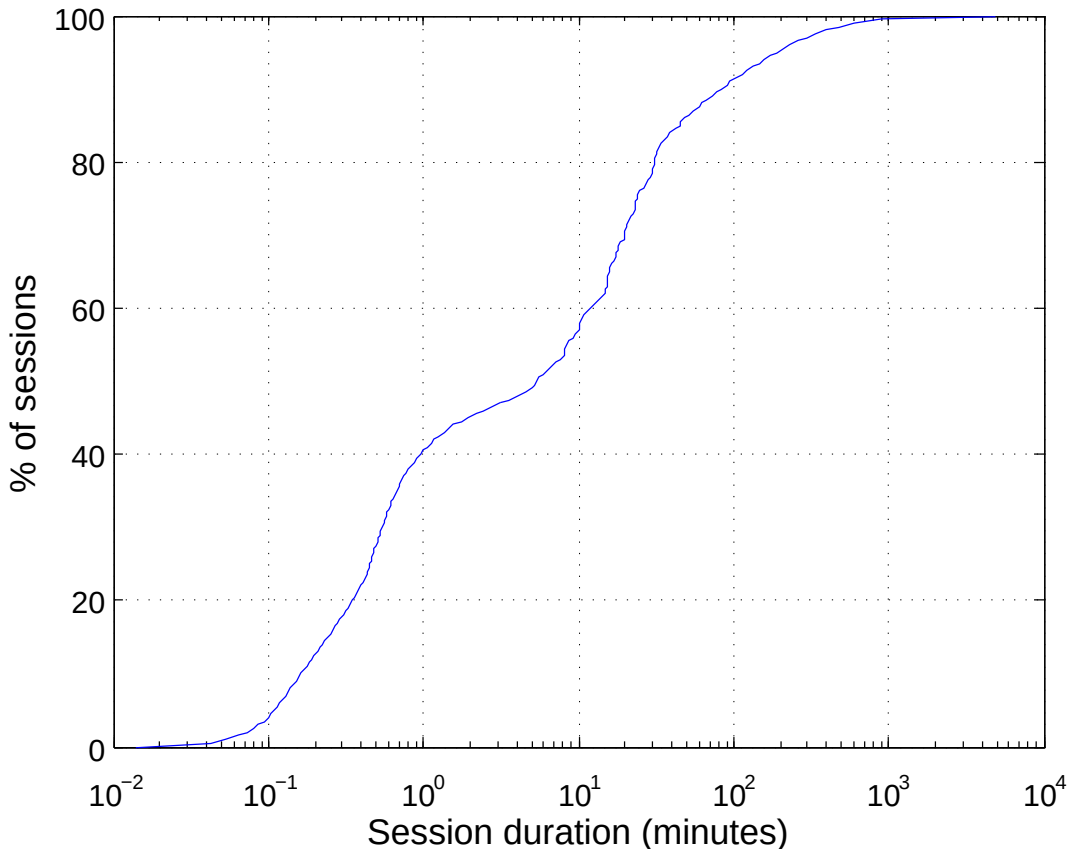


Figure 3.9: Distribution of C&C session durations.

3.4.1 Campaign datasets

Our study covers three spam campaigns summarized in Table 3.1. The "Pharmacy" campaign is a 26-day sample (19 active days) of an on-going Storm campaign advertising an on-line pharmacy. The "Postcard" and "April Fool" campaigns are two distinct and serial instances of self-propagation campaigns, which attempt to install an executable on the user's machine under the guise of being postcard software. For each campaign, Figure 3.7 shows the number of messages per hour assigned to bots for mailing.

Storm's authors have shown great cunning in exploiting the cultural and social expectations of users — hence the April Fool campaign was rolled out for a limited run around April 1st. Our Web site was designed to mimic the earlier Postcard campaign and thus our data probably does not perfectly reflect user behavior for this campaign, but the two are similar enough in nature that we surmise that any impact is small.

Table 3.2: The 10 most-targeted email address domains and their frequency in the combined lists of targeted addresses over all three campaigns.

DOMAIN	FREQ.
hotmail.com	8.47%
yahoo.com	5.05%
gmail.com	3.17%
aol.com	2.37%
yahoo.co.in	1.13%
sbcglobal.net	0.93%
mail.ru	0.86%
shaw.ca	0.61%
wanadoo.fr	0.61%
msn.com	0.58%
Total	23.79%

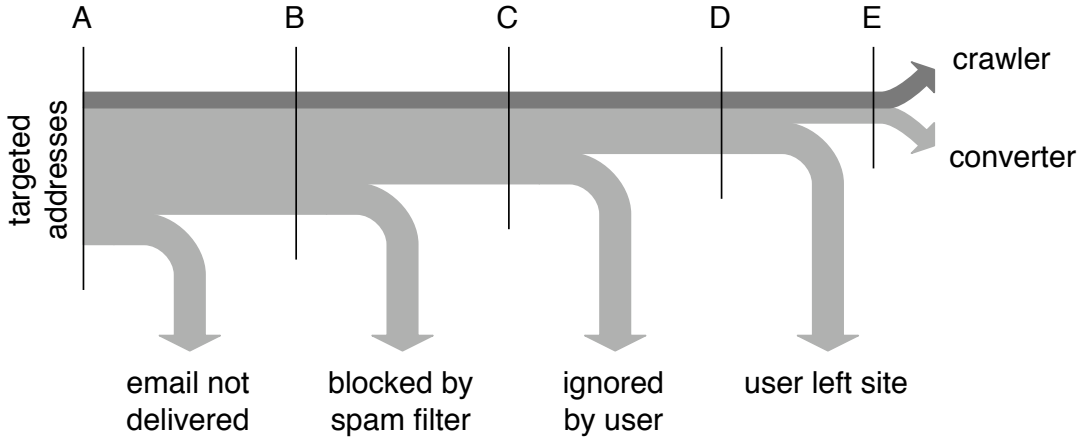


Figure 3.10: The spam conversion pipeline.

We began the experiment with 8 proxy bots, of which 7 survived until the end. One proxy crashed late on March 31. The total number of worker bots connected to our proxies was 75,869.

Figure 3.8 shows a timeline of the proxy bot workload. The number of workers connected to each proxy is roughly uniform across all proxies (23 worker bots on average), but shows strong spikes corresponding to new self-propagation campaigns. At peak, 539 worker bots were connected to our proxies at the same time.

Most workers only connected to our proxies once: 78% of the workers only

Table 3.3: Filtering at each stage of the spam conversion pipeline for the self-propagation and pharmacy campaigns. Percentages refer to the conversion rate relative to Stage A.

Stage	Pharmacy		Postcard		April Fool	
A – Spam Targets	347,590,389	100%	83,655,479	100%	40,135,487	100%
B – MTA Delivery (est.)	82,700,000	23.8%	21,100,000	25.2%	10,100,000	25.2%
C – Inbox Delivery	—	—	—	—	—	—
D – User Site Visits	10,522	0.00303%	3,827	0.00457%	2,721	0.00680%
E – User Conversions	28	0.0000081%	316	0.000378%	225	0.000561%

connected to our proxies a single time, 92% at most twice, and 99% at most five times. The most prolific worker IP address, a host in an academic network in North Carolina, USA, contacted our proxies 269 times; further inspection identified this as a NAT egress point for 19 individual infections. Conversely, most workers do not connect to more than one proxy: 81% of the workers only connected to a single proxy, 12% to two, 3% to four, 4% connected to five or more, and 90 worker bots connected to all of our proxies. On average, worker bots remained connected for 40 minutes, although over 40% workers connected for less than a minute. The longest connection lasted almost 81 hours.

The workers were instructed to send postcard spam to a total of 83,665,479 addresses, of which 74,901,820 (89.53%) are unique. The April Fool campaign targeted 38,651,124 addresses, of which 36,909,792 (95.49%) are unique. Pharmacy spam targeted 347,590,389 addresses, of which 213,761,147 (61.50%) are unique. Table 3.2 shows the 15 most frequently targeted domains of the three campaigns. The individual campaign distributions are identical in ordering and to a precision of one tenth of a percentage, therefore we only show the aggregate breakdown.

3.4.2 Spam conversion pipeline

Conceptually, we break down spam conversion into a pipeline with five "filtering" stages in a manner similar to that described by Aycock and Friess [4]. Figure 3.10 illustrates this pipeline and shows the type of filtering at each stage. The pipeline starts with delivery lists of target email addresses sent to worker bots (Stage A). For a wide range of reasons (e.g., the target address is invalid, MTAs refuse delivery because of blacklists, etc.), workers will successfully deliver only a subset of their messages to an

Table 3.4: Number of messages delivered to a user's inbox as a fraction of those injected for test accounts at free email providers and a commercial spam filtering appliance. The test account for the Barracuda appliance was not included in the Postcard campaign.

SPAM FILTER	PHARMACY	POSTCARD	APRIL FOOL
Gmail	0.00683%	0.00176%	0.00226%
Yahoo	0.00173%	0.000542%	none
Hotmail	none	none	none
Barracuda	0.131%	N/A	0.00826%

MTA (Stage B). At this point, spam filters at the site correctly identify many messages as spam, and drop them or place them aside in a spam folder. The remaining messages have survived the gauntlet and appear in a user's inbox as valid messages (Stage C). Users may delete or otherwise ignore them, but some users will act on the spam, click on the URL in the message, and visit the advertised site (Stage D). These users may browse the site, but only a fraction "convert" on the spam (Stage E) by attempting to purchase products (pharmacy) or by downloading and running an executable (self-propagation).

We show the spam flow in two parts, "crawler" and "converter", to differentiate between real and masquerading users (Section 3.3.4). For example, the delivery lists given to workers contain honeypot email addresses. Workers deliver spam to these honeypots, which then use crawlers to access the sites referenced by the URL in the messages (e.g., our own Spamscatter project [2]). Since we want to measure the spam conversion rate for actual users, we separate out the effects of automated processes like crawlers — a necessary aspect of studying an artifact that is also being actively studied by other groups [33].

Table 3.3 shows the effects of filtering at each stage of the conversion pipeline for both the self-propagation and pharmaceutical campaigns. The number of targeted addresses (A) is simply the total number of addresses on the delivery lists received by the worker bots during the measurement period, excluding the test addresses we injected.

We obtain the number of messages delivered to an MTA (B) by relying on delivery reports generated by the workers. Unfortunately, an exact count of successfully delivered messages is not possible because workers frequently change proxies or go offline, causing both extraneous (resulting from a previous, non-interposed proxy session) and missing

delivery reports. We can, however, estimate the aggregate delivery ratio (B/A) for each campaign using the success ratio of all observed delivery reports. This ratio allows us to then estimate the number of messages delivered to the MTA and even to do so on a per-domain basis.

The number of messages delivered to a user's inbox (C) is a much harder value to estimate. We do not know what spam filtering, if any, is used by each mail provider, and then by each user individually, and therefore cannot reasonably estimate this number in total. It is possible, however, to determine this number for individual mail providers or spam filters. The three mail providers and the spam filtering appliance we used in this experiment had a method for separating delivered mails into "junk" and inbox categories. Table 3.4 gives the number of messages delivered to a user's inbox for the free email providers, which together accounted for about 16.5% of addresses targeted by Storm (Table 3.2), as well as our department's commercial spam filtering appliance. It is important to note that these are results from one spam campaign over a short period of time and should not be used as measures of the relative effectiveness for each service. That said, we observe that the popular Web mail providers all do a very a good job at filtering the campaigns we observed, although it is clear they use different methods to get there (for example, Hotmail rejects most Storm spam at the MTA-level, while Gmail accepts a significant fraction only to filter it later as junk).

The number of visits (D) is the number of accesses to our emulated pharmacy and postcard sites, excluding any crawlers as determined using the methods outlined in Section 3.3.2. We note that crawler requests came from a small fraction of hosts but accounted for the majority of all requests to our Web sites. For the pharmacy site, for instance, of the 11,720 unique IP addresses seen accessing the site with a valid unique identifier, only 10.2% were blacklisted as crawlers. In contrast, 55.3% of all unique identifiers used in requests originated from these crawlers. For all non-image requests made to the site, 87.43% were made by blacklisted IP addresses.

The number of conversions (E) is the number of visits to the purchase page of the pharmacy site, or the number of executions of the fake self-propagation program.

Our results for Storm spam campaigns show that the spam conversion rate is quite low. For example, out of 350 million pharmacy campaign emails only 28 conversions

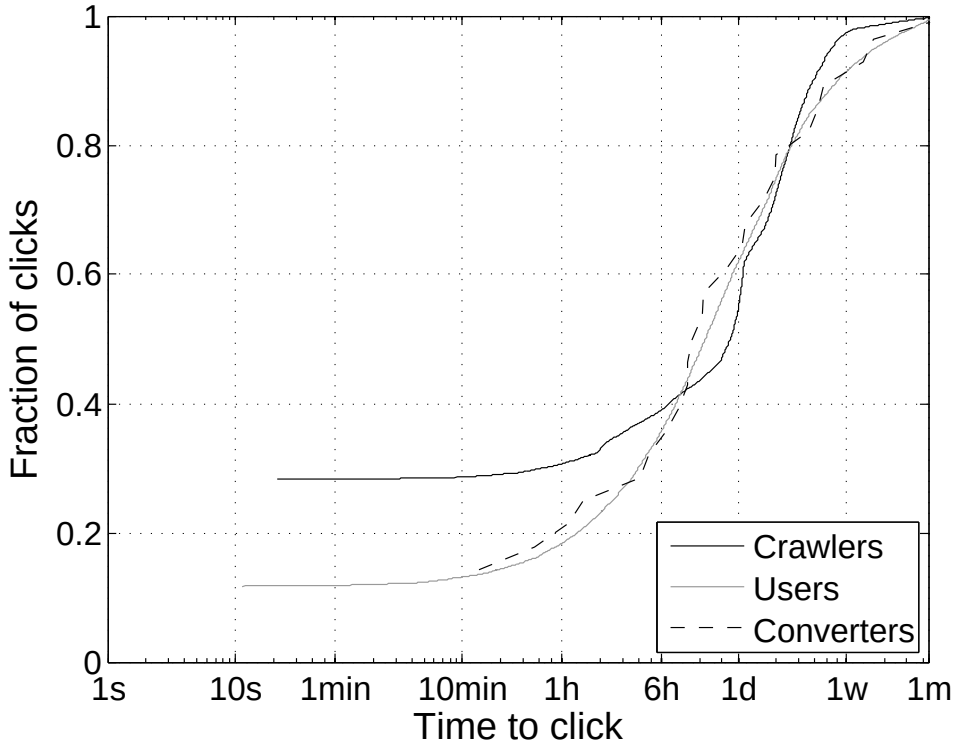


Figure 3.11: Time-to-click distributions for accesses to the pharmacy site.

resulted (and no crawler ever completed a purchase so errors in crawler filtering plays no role). However, a very low conversion rate does not necessary imply low revenue or profitability. We discuss the implications of the conversion rate on the spam conversion proposition further in Section 3.7.

3.4.3 Time to click

The conversion pipeline shows what fraction of spam ultimately resulted in visits to the advertised sites. However, it does not reflect the latency between when the spam was sent and when a user clicked on it. The longer it takes users to act, the longer the scam hosting infrastructure will need to remain available to extract revenue from the spam [2]. Put another way, how long does a spam-advertised site need to be available to collect its potential revenue?

Figure 3.11 shows the cumulative distribution of the "time-to-click" for accesses

to the pharmacy site. The time-to-click is the time from when spam is sent (when a proxy forwards a spam workload to a worker bot) to when a user "clicks" on the URL in the spam (when a host first accesses the Web site). The graph shows three distributions for the accesses by all users, the users who visited the purchase page ("converters"), and the automated crawlers (14,716 such accesses). Note that we focus on the pharmacy site since, absent a unique identifier, we do not have a mechanism to link visits to the self-propagation site to specific spam messages and their time of delivery.

The user and crawler distributions show distinctly different behavior. Almost 30% of the crawler accesses are within 20 seconds of worker bots sending spam. This behavior suggests that these crawlers are configured to scan sites advertised in spam immediately upon delivery. Another 10% of crawler accesses have a time-to-click of 1 day, suggesting crawlers configured to access spam-advertised sites periodically in batches. In contrast, only 10% of the user population accesses spam URLs immediately, and the remaining distribution is smooth without any distinct modes. The distributions for all users and users who "convert" are roughly similar, suggesting little correlation between time-to-click and whether a user visiting a site will convert. While most user visits occur within the first 24 hours, 10% of times-to-click are a week to a month, indicating that advertised sites need to be available for long durations to capture full revenue potential.

3.5 Effects of Blacklisting

A major effect on the efficacy of spam delivery is the employment by numerous ISPs of address-based blacklisting to reject email from hosts previously reported as sourcing spam. To assess the impact of blacklisting, during the course of our experiments we monitored the *Composite Blocking List* (CBL) [9], a blacklist source used by the operators of some of our institutions. At any given time the CBL lists on the order of 4–6 million IP addresses that have sent email to various spamtraps. We were able to monitor the CBL from March 21 – April 2, 2008, from the start of the Pharmacy campaign until the end of the April Fool campaign. Although the monitoring does not cover the full extent of all campaigns, we believe our results to be representative of the effects of CBL during the time frame of our experiments.

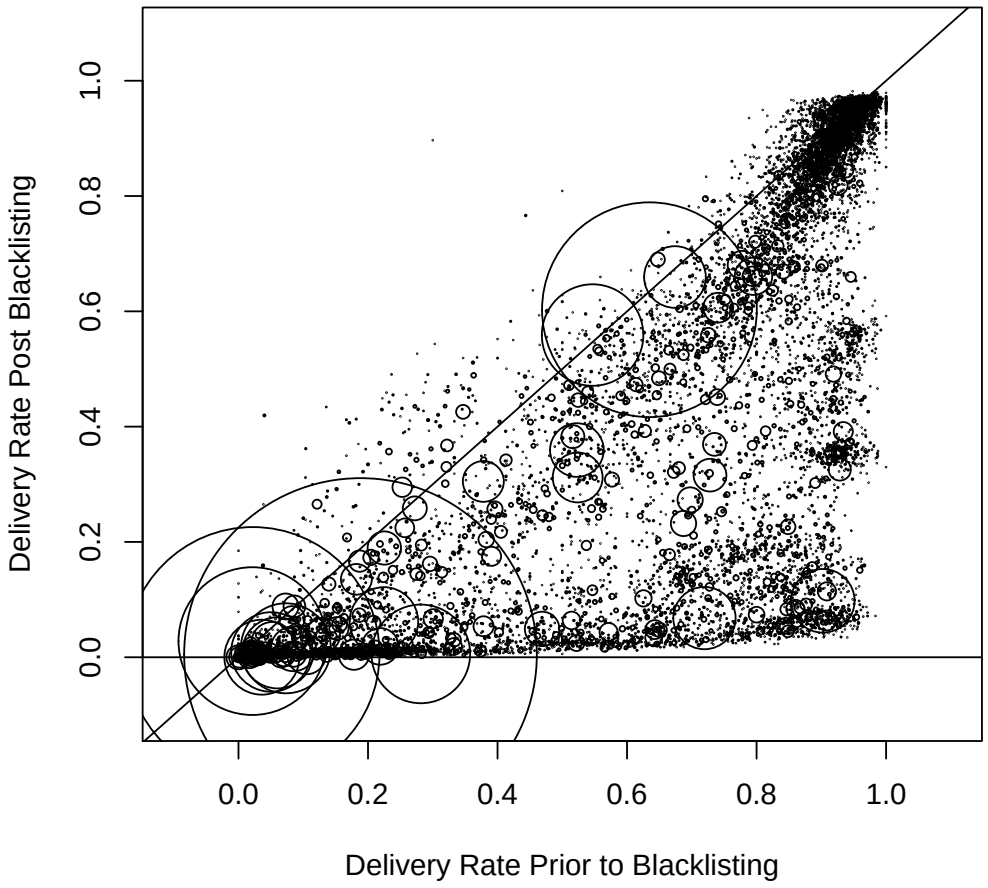


Figure 3.12: Change in per-domain delivery rates as seen prior to a worker bot appearing in the blacklist (x-axis) vs. after appearing (y-axis). Each circle represents a domain targeted by at least 1,000 analyzable deliveries, with the radius scaled in proportion to the number of delivery attempts.

We downloaded the current CBL blacklist every half hour, enabling us to determine which worker bots in our measurements were present on the list and how their arrival on the list related to their botnet activity. Of 40,864 workers that sent delivery reports, fully 81% appeared on the CBL. Of those appearing at some point on the list, 77% were on the list prior to our observing their receipt of spamming directives, appearing first on the list 4.4 days (median) earlier. Of those not initially listed but then listed subsequently, the median interval until listing was 1.5 hours, strongly suggesting that

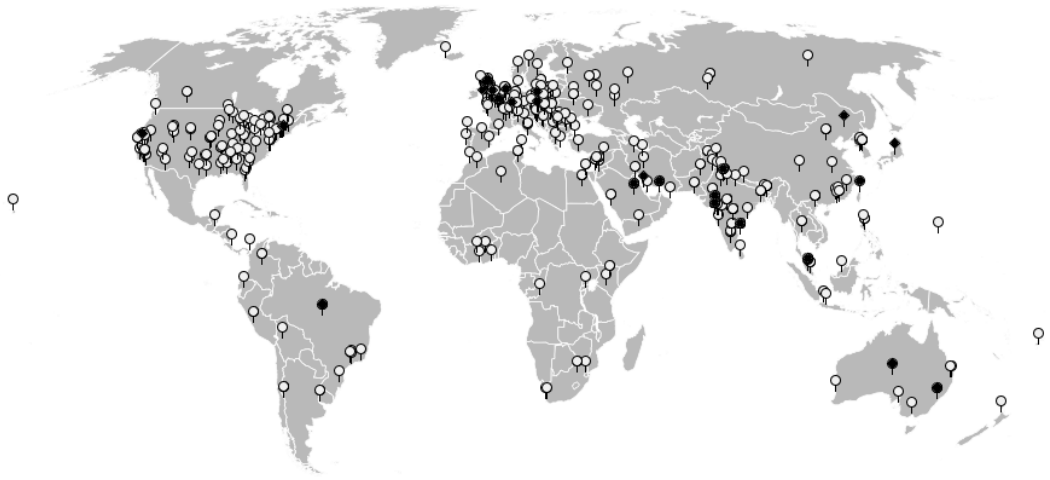


Figure 3.13: Geographic locations of the hosts that "convert" on spam: the 541 hosts that execute the emulated self-propagation program (light grey), and the 28 hosts that visit the purchase page of the emulated pharmacy site (black).

the spamming activity we observed them being instructed to conduct quickly led to their detection and blacklisting. Of hosts never appearing on the list, more than 75% never reported successful delivery of spam, indicating that the reason for their lack of listing was simply their inability to effectively annoy anyone.

One confounding factor is that the CBL exhibits considerable flux once an address first appears on the blacklist: the worker bots typically (median) experience 5 cycles of listing-followed-by-delisting. Much of this churn comes from a few periods of massive delistings, which appear to be glitches in maintenance (or propagation) of the blacklist rather than a response to external events. (If delistings arose due to botmasters using the delisting process to render their workers more effective for a while, then it might be possible to monitor the delisting process in order to conduct botnet counterintelligence, similar to that developed previously for blacklisting lookups [64].) Due to caching of blacklist entries by sites, we thus face ambiguity regarding whether a given worker is viewed as blacklisted at a given time. For our preliminary analysis, we simply consider a worker as blacklisted from the point where it first appears on the CBL onwards.

We would expect that the impact of blacklisting on spam delivery strongly depends on the domain targeted in a given email, since some domains incorporate blacklist feeds such as the CBL into their mailer operations and others do not. To explore this effect,

Figure 3.12 plots the per-domain delivery rate: the number of spam emails that workers reported as successfully delivered to the domain divided by number attempted to that domain. The x-axis shows the delivery rate for spams sent by a worker prior to its appearance in the CBL, and the y-axis shows the rate after its appearance in the CBL. We limit the plot to the 10,879 domains to which workers attempted to deliver at least 1,000 spams. We removed attempted deliveries for which we never saw a status report, as well as those for which the CBL listing appeared in between the directive to send the spam and the subsequent report, rendering it unclear whether the blacklist was in effect for the delivery. We also plot delivery rates for the two different campaigns as separate circles, though the overall nature of the plot does not change between them. The radius of each plotted circle scales in proportion to the number of delivery attempts, the largest corresponding to domains such as hotmail.com, yahoo.com, and gmail.com.

From the plot we clearly see a range of blacklisting behavior by different domains. Some employ other effective anti-spam filtering, indicated by their appearance near the origin — spam did not get through even prior to appearing on the CBL blacklist. Some make heavy use of either the CBL or a similar list (y-axis near zero, but x-axis greater than zero), while others appear insensitive to blacklisting (those lying on the diagonal). Since points lie predominantly below the diagonal, we see that either blacklisting or some other effect related to sustained spamming activity (e.g., learning content signatures) diminishes the delivery rate seen at most domains. Delisting followed by relisting may account for some of the spread of points seen here; those few points above the diagonal may simply be due to statistical fluctuations. Finally, the cloud of points to the upper right indicates a large number of domains that are not much targeted individually, but collectively comprise a significant population that appears to employ no effective anti-spam measures.

3.6 Conversion analysis

We now turn to a preliminary look at possible factors influencing response to spam. For the present, we confine our analysis to coarse-grained effects.

We start by mapping the geographic distribution of the hosts that “convert” on

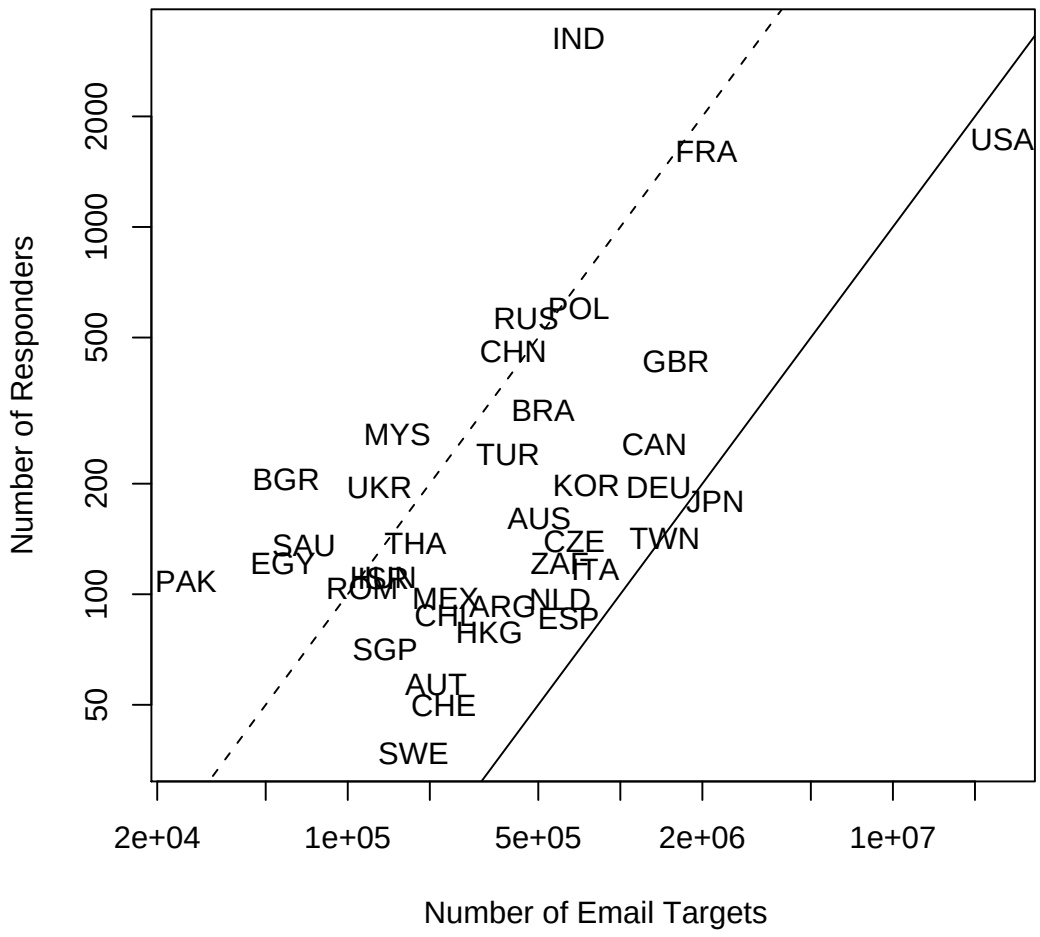


Figure 3.14: Volume of email targeting (x-axis) vs. responses (y-axis) for the most prominent country-code TLDs. The x and y axes correspond to Stages A and D in the pipeline (Figure 3.10), respectively.

the spam campaigns we monitored. Figure 3.13 maps the locations of the 541 hosts that execute the emulated self-propagation program, and the 28 hosts that visit the purchase page of the emulated pharmacy site. The map shows that users around the world respond to spam.

Figure 3.14 looks at differences in response rates among nations as determined by prevalent country-code email domain TLDs. To allow the inclusion of generic TLDs such as `.com`, for each email address we consider it a member of the country hosting its

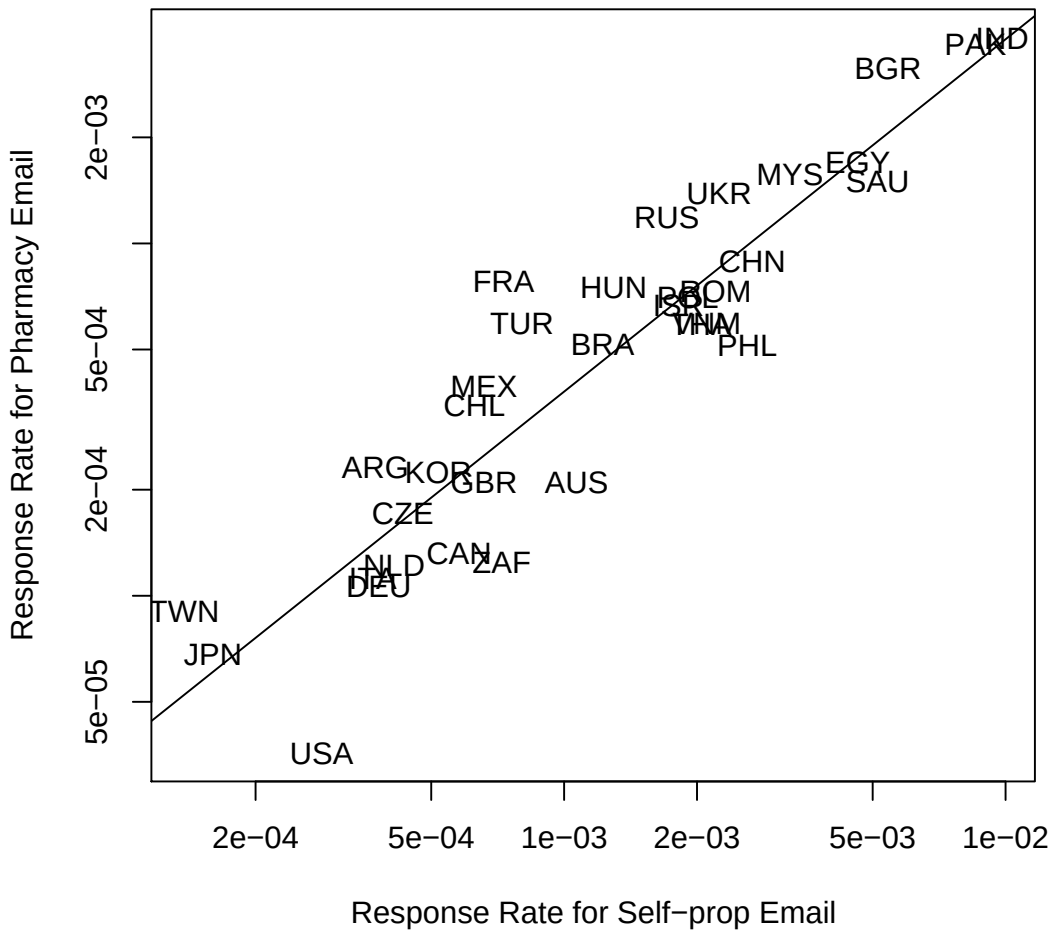


Figure 3.15: Response rates (stage D in the pipeline) by TLD for executable download (x-axis) vs. pharmacy visits (y-axis).

mail server; we remove domains that resolve to multiple countries, categorizing them as "international" domains. The x-axis shows the volume of email (log-scaled) targeting a given country, while the y-axis gives the number of responses recorded at our Web servers (also log-scaled), corresponding to Stages A and D in the pipeline (Figure 3.10), respectively. The solid line reflects a response rate of 10^{-4} and the dashed line a rate of 10^{-3}. Not surprisingly, we see that the spam campaigns target email addresses in the United States substantially more than any other country. Further, India, France and the United States dominate responses. In terms of response *rates*, however, India, Pakistan,

and Bulgaria have the highest response rates than any other countries (furthest away from the diagonal). The United States, although a dominant target and responder, has the lowest resulting response rate of any country, followed by Japan and Taiwan.

However, the countries with predominant response rates do not appear to reflect a heightened interest in users from those countries in the specific spam offerings. Figure 3.15 plots the rates for the most prominent countries responding to self-propagation vs. pharmacy spams. The median ratio between these two rates is 0.38 (diagonal line). We see that India and Pakistan in fact exhibit almost exactly this ratio (upper-right corner), and Bulgaria is not far from it. Indeed, only a few TLDs exhibit significantly different ratios, including the US and France, the two countries other than India with a high number of responders; users in the US respond to the self-propagation spam substantially more than pharmaceutical spam, and vice-versa with users in France. These results suggest that, for the most part, per-country differences in response rate are due to *structural* causes (quality of spam filtering, general user anti-spam education) rather than differing degrees of cultural or national interest in the particular promises or products conveyed by the spam.

3.7 Summary

This chapter describes what we believe is the first large-scale quantitative study of spam conversion. We developed a methodology that uses botnet infiltration to indirectly instrument spam emails such that user clicks on these messages are taken to replica Web sites under our control. Using this methodology we instrumented almost 500 million spam messages, comprising three major campaigns, and quantitatively characterized both the delivery process and the conversion rate.

We would be the first to admit that these results represent a single data point and are not necessarily representative of spam as a whole. Different campaigns, using different tactics and marketing different products will undoubtedly produce different outcomes. Indeed, we caution *strongly* against researchers using the conversion rates we have measured for these Storm-based campaigns to justify assumptions in any other context. At the same time, it is tempting to speculate on what the numbers we have

measured might *mean*. We succumb to this temptation below, with the understanding that few of our speculations can be empirically validated at this time.

After 26 days, and almost 350 million email messages, only 28 sales resulted — a conversion rate of well under 0.00001%. Of these, all but one were for male-enhancement products and the average purchase price was close to $100. Taken together, these conversions would have resulted in revenues of $2,731.88 — a bit over $100 a day for the measurement period or $140 per day for periods when the campaign was active. However, our study interposed on only a small fraction of the overall Storm network — we estimate roughly 1.5 percent based on the fraction of worker bots we proxy. Thus, the total daily revenue attributable to Storm's pharmacy campaign is likely closer to $7000 (or $9500 during periods of campaign activity). By the same logic, we estimate that Storm self-propagation campaigns can produce between 3500 and 8500 new bots per day.

Under the assumption that our measurements are representative over time (an admittedly dangerous assumption when dealing with such small samples), we can extrapolate that, were it sent continuously at the same rate, Storm-generated pharmaceutical spam would produce roughly 3.5 million dollars of revenue in a year. This number could be even higher if spam-advertised pharmacies experience repeat business. A bit less than "millions of dollars every day", but certainly a healthy enterprise.

The next obvious question is, "How much of this revenue is profit"? Here things are even murkier. First, we must consider how much of the gross revenue is actually recovered on a sale. Assuming the pharmacy campaign drives traffic to an affiliate program (and there are very strong anecdotal reasons to believe this is so) then the gross revenue is likely split between the affiliate and the program (a annual net revenue of $1.75M using our previous estimate). Next, we must subtract business costs. These include a number of incidental expenses (domain registration, bullet-proof hosting fees, etc) that are basically fixed sunk costs, and the cost to distribute the spam itself.

Anecdotal reports place the *retail* price of spam delivery at a bit under $80 per million [84]. This cost is an order of magnitude less than what legitimate commercial mailers charge, but is still a significant overhead; sending 350M emails would cost more than $25,000. Indeed, given the net revenues we estimate, retail spam delivery would

only make sense if it were 20 times cheaper still.

And yet, Storm continues to distribute pharmacy spam — suggesting that it is in fact profitable. One explanation is that Storm's masters are vertically integrated and the purveyors of Storm's pharmacy spam are none other than the operators of Storm itself (i.e., that Storm does not deliver these spams for a third-part in exchange for a fee). There is some evidence for this, since the distribution of target email domain names between the self-propagation and pharmacy campaigns is virtually identical. Since the self-propagation campaigns fundamentally must be run by the botnet's owners, this suggests the purveyor of the pharmacy spam is one and the same. A similar observation can be made in the harvesting of email addresses from the local hard drives of Storm hosts. These email addresses subsequently appear in the target address lists of the pharmacy campaign and self-propagation campaigns alike. Moreover, neither of these behaviors is found in any of the other (smaller) campaigns distributed by Storm (suggesting that these may in fact be fee-for-service distribution arrangements). If true, then the cost of distribution is largely that of the labor used in the development and maintenance of the botnet software itself. While we are unable to provide any meaningful estimates of this cost (since we do not know which labor market Storm is developed in), we surmise that it is roughly the cost of two or three good programmers.

If true, this hypothesis is heartening since it suggests that the third-party *retail* market for spam distribution has not grown large or efficient enough to produce competitive pricing and thus, that profitable spam campaigns require organizations that can assemble complete "soup-to-nuts" teams. Put another way, the profit margin for spam (at least for this one pharmacy campaign) may be meager enough that spammers must be sensitive to the details of how their campaigns are run and are economically susceptible to new defenses.

This chapter has presented a very deep analysis of the operation of one spamming botnet, allowing us to understand the volume of spam sent, conversion rate of those messages, and make rough estimates of one spammer's gross revenue. The next chapter broadens our approach, allowing us to investigate multiple spammers, and get a better idea both of the demand for their products and the revenues of the spam-advertised goods industry at large.

Chapter 3, in part, is a reprint of the material as it appears in Proceedings of ACM CCS 2008. Kanich, Chris; Kreibich, Christian; Levchenko, Kirill; Enright, Brandon; Voelker, Geoffrey M.; Paxson, Vern; Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, is a reprint of the material as it appears in Proceedings of IMC 2011. Kreibich, Christian, Weaver, Nicholas; Kanich, Chris; Cui, Weidong; Paxson, Vern.

Chapter 4

Understanding the Business Processes of Unsolicited Commercial Email Merchants

The previous chapter presents an in depth characterization of the spam campaigns sent by an individual botnet. Yet this is but an individual spammer's operation within, presumably, a much broader ecosystem. This chapter presents research which widens our understanding of the market for spam advertised goods both on the demand side, by analyzing the distribution of goods desired by end users of a spam advertised pharmacy, and on the supply side, through inferring the order throughput of several large affiliate marketing programs.

4.1 Introduction

A large number of Internet scams are "advertising-based"; that is, their goal is to convince potential customers to purchase a product or service, typically via some broad-based advertising medium.[1] In turn, this activity mobilizes and helps fund a broad array of technical capabilities, including botnet-based distribution, fast flux name service,

[1] Unauthorized Internet advertising includes email spam, black hat search-engine optimization [82], blog spam [57], Twitter spam [21], forum spam, and comment spam. Hereafter we refer to these myriad advertising vectors simply as spam.

and bulletproof hosting. However, while these same technical aspects enjoy a great deal of attention from the security community, there is considerably less information quantifying the underlying economic engine that drives this ecosystem. Absent grounded empirical data, it is challenging to reconcile revenue "estimates" that can range from $2M/day for one spam botnet [1], to analyses suggesting that spammers make little money at all [25]. This situation has the potential to distort policy and investment decisions that are otherwise driven by intuition rather than evidence.

On the demand side, as expected, we find that most pharmaceuticals selected for purchase are in the "male-enhancement" category (primarily Viagra and other erectile dysfunction medications comprising 60 distinct items). However, such drugs constitute only 62% of the total, and we document that this demand distribution has quite a long tail; user shopping carts contain 289 distinct products, including surprising categories such as anti-cancer medications (Arimidex and Gleevec), anti-schizophrenia drugs (Seroquel), and asthma medications (Advair and Ventolin). We also discover significant differences in the purchasing habits of U.S. and non-U.S. customers.

Combining these measurements, we synthesize overall revenue estimates for each program, which can be well in excess of $1M per month for a single enterprise. To the best of our knowledge, ours is the first empirical data set of its kind, as well as the first to provide insight into the market size of the spam-advertised goods market and corresponding customer purchasing behavior.

The remainder of this chapter is structured as follows. We discuss our *purchase pair* technique in Section 4.2, validating our technique for internal consistency and then presenting order volume estimates across seven of the top pharmaceutical affiliate programs and three counterfeit software programs. We then explore the customer dynamics for one particular pharmaceutical program, EvaPharmacy, in Section 4.3. We explain how to use image log data to identify customer purchases and then document how, where and when the EvaPharmacy customer base places its orders. We summarize our findings in Section 4.4, devising estimates of revenue and comparing them with external validation. We conclude with a discussion about the implications of our findings in Section 4.5.

4.2 Order volume

Underlying our *purchase pair* measurement approach is a model of how affiliate programs handle transactions, and, in particular, how they assign order numbers.

4.2.1 Basic idea

Upon placing an order, most affiliate programs provide a confirmation page that includes an "order number" (typically numeric, or at least having a clear numeric component) that uniquely specifies the customer's transaction. For purchases where an order number does not appear on the confirmation page, the seller can provide one in a confirmation email (the common case), or make one available via login to the seller's Web site. The order number allows the customer to specify the particular purchase in any subsequent emails, when using customer support Web sites, or when contacting online support via email, IM or live Web chat. For the purchases we made, we found that the seller generally provides the order number *before* the authorization step (indeed, even before merchant-side fraud checks such as Address Verification Service), although purely local checks such as Luhn digit validation are frequently performed first. Accordingly, we can consider the creation of an order number only as evidence that a customer *attempted* an order, not that it successfully concluded. Thus, the estimates we form in this work reflect an *upper bound* on the transaction rate, including transactions declined during authorization or settlement.[2]

The most important property for such order numbers is their *uniqueness*; that each customer order is assigned a singular number that is distinguished over time without the possibility of aliasing. While there are a vast number of ways such uniqueness could be implemented (e.g., a pseudo-random permutation function), the easiest approach by far is to simply increment a global variable for each new order. Indeed, the serendipitous observation that motivated our study was that multiple purchases made from the same affiliate program produced order numbers that appeared to *monotonically increase* over

[2] In 2008, Visa documented that card-not-present transactions such as e-commerce had an issuer decline rate of 14% system-wide [81]. In addition, it seems likely that some orders are declined at the merchant's processor due to purely local fraud checks (such as per-card or per-address velocity checks or disparities between IP address geolocation versus shipping address).

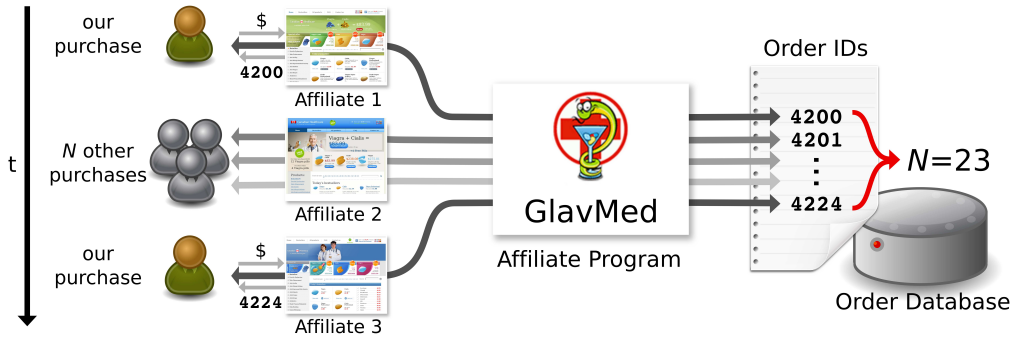


Figure 4.1: How the purchase pair technique works. In this hypothetical situation, two measurement purchases are made that bracket some number of intervening purchases made by real customers. Because order number allocation is implemented by a serialized sequential increment, the difference in the order numbers between measurement purchases, $N = 23$, corresponds to the total number of orders processed by the affiliate program in the intervening time.

time. Observing the monotonic nature of this sequence, we hypothesized that order number allocation is implemented by serializing access to a single global variable that is incremented each time an order is made; we call this the *sequential update hypothesis*. To assess this hypothesis, we examined source code for over a dozen common e-commerce platforms (e.g., Magento, X-cart, Ubercart, and Zen-cart [49, 79, 86, 87]), finding ubiquitous use of such a counter, typically using an SQL auto-update field, but sometimes embodied explicitly in code.

Given use of such a global sequential counter, the *difference* between the numbers associated with orders placed at two points in time reflects the total number of orders placed during the intervening time period. Thus, from any *pair* of purchases we can extract a measurement of the total transaction volume for the interval of time between them, even though we cannot directly witness those intervening transactions. Figure 4.1 illustrates the methodology using a concrete example. This observation is similar in flavor to the analysis used in blind/idle port scanning (there the sequential increment of the IP identification field allows inference of the presence of intervening transmissions) [11]. It then appears plausible that this same purchase-pair approach might work across a broad range of spam-advertised programs, a possibility that we explore more thoroughly next.

4.2.2 Data collection

To evaluate this approach requires that we first identify which sites advertise which affiliate programs, and then place repeated purchases from each. We describe how we gathered each of these data sets in this section.

Program data

In prior work, we developed a URL crawler to follow the embedded links contained in real-time feeds of email spam (provided by a broad range of third-party anti-spam partners) [46]. The crawler traverses any redirection pages and then fetches and renders the resulting page in a live browser. We further developed a set of "page classifiers" that identify the type of good being advertised by analyzing the site content, and, in most cases, the particular affiliate program being promoted. We developed specific classifiers for over 20 of the top pharmaceutical programs (comprising virtually all sites advertised in pharmaceutical spam), along with the four most aggressively spam-advertised counterfeit software programs.

After placing multiple test orders with nine of these pharmaceutical programs, we identified seven with strictly incrementing order numbers.[3] Five of these (Rx–Promotion, Pharmacy Express (aka Mailien), GlavMed, Online Pharmacy and EvaPharmacy) together constituted two-thirds of all sites advertised in the roughly 350 million distinct pharmaceutical spam URLs we observed over three months in late 2010. We found the sixth, 33drugs (aka DrugRevenue), and seventh, 4RX, less prevalent in email spam URLs, but they appear to be well advertised via search engine optimization (SEO) techniques [44]. We did a similar analysis of counterfeit software programs, finding three (Royal Software, EuroSoft, and SoftSales) with the appropriate order-number signature. While counterfeit software is less prevalent in total spam volume, these three programs constitute over 97% of such sites advertised to our spam collection apparatus during the same 3-month period. For the remainder of this paper we focus exclusively on these ten programs, although it appears plausible that the same technique will prove applicable to

[3] Of the two programs that we did not select, ZedCash used several different strictly increasing order number subspaces that would complicate our analysis and decrease accuracy, while World Pharmacy order numbers appeared to be the concatenation of a small value with the current Unix timestamp, which would thwart our analysis altogether.

many smaller programs, and also to programs in other such markets (e.g., gambling, fake antivirus, adult).

Order data

We collected order data in two manners: actively via our own purchases and opportunistically, based on the purchases of others. First and foremost are our own purchases, which we conducted in two phases. The first phase arose during a previous study, during which we executed a small number of test purchases from numerous affiliate programs in January and November of 2010 using retail Visa gift cards. Of these, 46 targeted the ten programs under study in this paper. The second phase (comprising the bulk of our active measurements) reflects a regimen of purchases made over three weeks in January and February 2011 focused specifically on the ten programs we identified above.

When placing these orders, we used multiple distinct URLs leading to each program (as identified by our page classifiers). The goal of this procedure was to maximize the likelihood of using distinct affiliates to place purchases in order to provide an opportunity to determine whether different affiliates of a given program make use of different order-processing services.

Successfully placing orders had its own set of operational challenges outlined in Section 4.2.3. Except where noted, we performed all of our purchases using prepaid Visa credit cards provided to us in partnership with a specialty issuer, and funded to cover the full amount of each transaction. We used a distinct card for each purchase and went to considerable lengths to emulate real customers. We used valid names and associated residential shipping addresses, placed orders from a range of geographically proximate IP addresses, and provided a unique email address for each order. We used five contact phone numbers for order confirmation, three from Google Voice and two via prepaid cell phones, with all inbound calls routed to the prepaid cell phones. In a few instances we found it necessary to place orders from IP addresses closely geolocated to the vicinity of the billing address for a given card, as the fraud check process for one affiliate program (EuroSoft) was sensitive to this feature. Another program (Royal Software) would only accept one order per IP address, requiring IP address diversity as well.

In total we placed 156 such orders. We scheduled them both periodically over a three-week period as well as in patterns designed to help elucidate more detail about transaction volume and to test for internal consistency, as discussed below.

Finally, in addition to the raw data from our own purchase records, we were able to capture several purchase order numbers via forum scraping. This opportunity arose because affiliate programs typically sponsor online forums that establish a community among their affiliates and provide a channel for distributing operational information (e.g., changes in software or name servers), sharing experiences (e.g., which registrars will tolerate domains used to host pharmaceutical stores), and to raise complaints or questions. One forum in particular, for the GlavMed program, included an extended "complaint" thread in which individual affiliates complained about orders that had not yet cleared payment processing (important to them since affiliates are only paid for each settled transaction that they deliver). These affiliates chose to document their complaints by listing the order number they were waiting for, which we determined was in precisely the same format and numeric range as the order numbers presented to purchasers. By mining this forum we obtained 122 numbers for past orders, including orders dating back to 2008.

Note that this data contains an innate time bias since the date of complaint inevitably came a while later than the time of purchase (unlike our own purchases). For this reason, we identify opportunistically gathered points distinctly when analyzing the data. We will see below that the bias proves to be relatively minor.

We summarize the total data set in Table 4.1. It includes order numbers from 202 active purchases and 122 opportunistically gathered data points.

4.2.3 Purchasing Methodology

While some studies can be completed purely using network-level measurements (either active or passive), in many cases this vantage point can only take one so far. In particular, when studying the nature of goods and services on offer via the criminal ecosystem (e.g., including those advertised to the general public, such as spam-advertised pharmaceuticals, and those rendered to "the trade" such as underground VPNs, exploit kits, compromised accounts and so on) it is difficult to do so without placing direct

Table 4.1: Active orders placed to sites of each affiliate program in the two different time phases of our study. In addition, we opportunistically gathered 122 orders for GlavMed covering the period between 2/08 and 1/11.

Affiliate Program	Phase 1 (1/10 – 11/10)	Phase 2 (1/11 – 2/11)
Rx–Promotion	7	27
Pharmacy Express	3	9
GlavMed	12	14
Online Pharmacy	5	16
EvaPharmacy	7	16
33drugs	4	16
4RX	1	13
EuroSoft	3	25
Royal Software	2	9
SoftSales	2	11

financial transactions via purchasing.

Placing such orders can be operationally difficult, however, and ensuring "realism" creates particular challenges. In this section we explain these challenges and how our protocol for handling financial transactions has changed over the last two years of active involvement.

First, we should make clear that independent of the challenges to verisimilitude, active measurements such as purchasing from criminals create their own ethical, legal and operational sensitivities. These issues are not the focus of this paper, but we wish to emphasize that they have consumed significant attention. All of the work we describe has been with the knowledge and oversight of multiple lawyers—both specialists in cyberlaw and general counsel for our institution—and has either been reviewed by our IRB (when they deemed human subjects to be involved) or consistent with a pre-established set of ethical guidelines that our group has followed consistently. Indeed, we invested significant time in consultation with, and education of, our administrators, overseers and advisers, to arrive at these decisions. Finally, managing the *funding* of such activities through a university administration took several years of internal trust building—"We need to be reimbursed for large numbers of cash equivalent payments for goods that may never be delivered, that will probably have no receipt and will, at times, involve our being

defrauded. Is that okay?"—and the development of appropriate industry funding sources.

The remainder of this section focuses on the *operational* requirements of such purchasing activity. We separate these financial interactions into two categories: those in which we pose as fellow scammers and those in which we pose as customers from the general public. The distinction is driven both by the unique characteristics of each domain as well as the requirements needed to maintain "cover" in taking measurements.

Purchasing as a customer

Purchasing goods or services offered for sale via traditional channels (e.g., using credit cards via Web sites) appears far easier on the surface, but introduces its own unique challenges in execution at scale. We purchased from a large selection of pharmaceutical, software, and counterfeit luxury goods affiliate programs to identify critical elements of the payment infrastructure [46]; for this effort, we attempted 120 purchases totaling $10,400 over the course of one month in 2010. For the research presented in this chapter, we attempted 156 purchases over three weeks totaling $6,600.

Payment cards

The first challenge is in finding appropriate payment instruments. While it is easy to use one's own personal credit card to place a order for spam-advertised herbal supplements or for SEO-advertised fake anti-virus software, this approach has many drawbacks. First it exposes researchers to potential fraud by providing their credit card information. Second, it is difficult to differentiate between orders on a credit card statement since the merchant's identification string is frequently unrelated to the name of the Web site at which an order was placed. Finally, placing many orders from the same card creates a suspicious profile and will quickly trigger standard velocity checks in the merchant's (or issuer's) payment fraud system (or, more likely, a fraud check system operated by their payment processor or gateway service provider).

One alternative that seems to address this issue is the prepaid gift card (issued by banks through both the Visa and MasterCard associations). Such cards are practically anonymous (since the card holder's identity is bound late, only after the card is purchased, with zero due diligence) and are cheap enough that different cards can be used for

different transactions (albeit with some loss due to the residual balance not being easily transferable).[4] Moreover, some issuers will provision "virtual" cards online, allowing new cards to be created on demand and with variable amounts. However, after using such cards for almost eighteen months, we can report that they are less attractive than they first appear.

First, most gift cards only provide telephone support. The assumption is that most users care primarily about their gift card balance (which can be provided via an automated telephone menu) and all other requests (e.g., finding out if a particular transaction settled, the date it settled on, the claimed merchant ID, etc.) require speaking with the customer support desk. Our experience is that even in widely-used gift card brands the customer support desk is staffed by only a few individuals and calling tens of times to request further information creates suspicion of fraud. Moreover, other key information (e.g., information about transactions that authorized but did not settle and the Acquirer's Reference Number, or ARN, identifying the acquiring bank used by the merchant) does not appear to be available to most support desk operators.

A subset of gift cards provides a Web interface that allows holders to obtain most of this information online, thus avoiding the customer support problem.[5] However, after researching the Visa gift card market extensively, we found extremely few U.S. issuers who provide an online Web interface that includes ARN information (critical for experiments that seek to cluster payment processing infrastructure used by different scammers). Through consulting with another researcher in the field, we were directed to one particular brand of card—sold over the counter in West Coast supermarkets—having these properties. We purchased several thousand dollars of these cards only to be undone by Federal legislation. As part of the Credit Card Reform Act of 2009, the FinCEN division of the Treasury department was mandated to revise regulations on the use of prepaid credit cards to address their use as money laundering vehicles. Among these new rules (first proposed in mid-2010) are strict reporting requirements on international

[4] In our experience, placing a single purchase per card number is highly preferable, both for associating individual transactions to individual cards as well as for analyzing future fraud actions. In initial experiments we used $500 gift cards and made multiple purchases on each card, only to find that we could not determine the source of subsequent fraudulent transactions because we had used the card at a variety of sites, making the culprit ambiguous. In our current efforts, we use each credit card number only a single time.

[5] Note that there is no guarantee of such service and one such provider changed their policy *after* we had purchased their cards, returning us to telephone support for key information.

transfers. In response, most U.S. Visa gift card issuers elected to simply restrict their cards to domestic transactions. This change was problematic for cybercrime research since virtually all interesting transactions are settled through foreign banks.

In the end, after two years of experimentation, we do not believe there is an easy solution to this problem that is broadly available. Our ultimate solution was to contract directly with a specialty card issuer whose products do not constitute a "prepaid program" and who agreed explicitly to support our research, issue new card numbers on demand (our protocol is to use a single credit card number only one time), and manually export fine-grained information on each transaction (authorization, settlement, acquiring Bank Identification Number (BIN), Card Acceptor ID (CAID), etc.) for a nominal fee. This method required significant negotiation effort on our part as well as an initial investment of $16,000. While it has been tremendously valuable for us, it is unfortunately non-trivial to replicate.

Online fraud checks

Armed with a large supply of Visa payment cards we quickly found that many of our purchases were declined by the sites we purchased from. Through a combination of trial and error (as well as research into commercial payment fraud services) we discovered that many merchants employ a range of anti-fraud measures that we had not anticipated.

The first is the standard Address Verification System (AVS), which is provided through the card association and validates the numeric portion of the customer's street address as well as their ZIP code. In each purchase we had inadvertently forgotten to correctly "program" each card with the corresponding shipping address we planned to use and our transactions were nearly always declined as a result.

Having fixed the address issue, we still found a variety of sites that rejected our orders, typically stating that the "fraud score" was above a threshold. Based on our experiments and examining third-party "fraud check" services, we believe these checks include IP geolocation (matching the location of the purchaser with the shipping address and AVS), validating that shipping addresses correspond to "residential" locations, and flagging "free" email accounts (which, being free, are considered riskier).

For such sites, we modified our operational protocol to address all three issues.

We obtained IP endpoints located close to the associated shipping addresses, we switched to using residential shipping addresses instead of the commercial mailbox provider we had used initially, and we created a range of new domain names to source email addresses (using Google Apps to host the underlying email service). While this significantly reduced our decline rate, we still found sites that tracked past IP purchase history and we needed to allocate a unique IP address for each purchase at such sites.[6] Finally, as with our use of payment cards, we learned that there was significant value in using a unique email address for each purchase—doing so allows disambiguation among customer service messages, and tracking differences in post-order advertising.

Voice contact

Most scam sites require the purchaser to provide a range of contact information, not only shipping and email addresses, but a voice phone number as well—all to support their own fraud concerns. In our experience, these voice numbers get used frequently to confirm orders (as yet another fraud check). This in turn requires a range of phone numbers and some way to relate caller to orders (to recall which "identity" is being called by the merchant). Over time we have adopted a system using multiple prepaid cellular telephones, each associated with multiple Google Voice accounts. This arrangement allows for easy centralized access to voicemail for each account, but also permits the use of geographically accurate phone numbers. We maintained an online spreadsheet that identified outstanding orders, as well as the associated names, credit cards and phone numbers, allowing our purchasers to quickly determine what an incoming call might be related to.

Managing the operations of this channel was among the most problematic of our efforts. First, the time overhead of playing phone tag with different merchants can add up over hundreds of orders. Moreover, even using our shared spreadsheet it was not always clear which order was being called about and our group members were forced to bluff their way through conversations until they could determine their appropriate identity. A further challenge, similar to our experience with gift card support, is that over

[6] Note that a range of sites will block Tor exit nodes and thus Tor is not an effective solution to this problem.

time phone operators would come to recognize our voices, requiring different "actors" to handle such calls. Finally, if our purchasing behavior exceeded a certain level (at one point we inadvertently placed a large number of orders for an identical product to the same address) it was via phone that we would be challenged, further placing pressure on our buyers to "think fast" for an explanation. When we failed at such ruses, we found that all related orders would be blacklisted (presumably using some combination of source IP address and shipping address).

Shipping

Finally, while virtual goods (e.g., fake anti-virus software, counterfeit software, malware, etc.) can be delivered online, physical goods must be shipped to a postal address. Purchasing these physical goods creates a number of challenges: since we use a range of distinct names for placing orders, there must be associated postal addresses that will accept mail for those individuals.

A natural concern when making these purchases is whether any goods would be received at all. In fact, similar to other online businesses, our experience is that customer service is prioritized and all orders were fulfilled (with a few exceptions due to our own errors).

Our first approach was to deliver all packages to a "virtual suite" at a rented commercial mailbox in a postal annex. Under this arrangement, we simply provided the postal annex with a list of names and then one of us was permitted to pickup shipments for any of those names. Over time this approach created multiple points of stress. First, while products shipped within the U.S. (i.e., via the postal service) did not require a signature on delivery, international shipments typically did. In the beginning the annex employees would allow a single individual to sign for these packages, but as volume increased they demanded signatures from each recipient, creating a bottleneck. Moreover, it became challenging to associate each individual product with the associated order (since it was common for the packaging to not identify the seller or the particular order number provided via the site's payment page). We also believe that our use of a non-residential address increased our fraud score for some merchant's payment systems, increasing the probability of a decline. Finally, at least one seller began to notice the

range of purchases to different names being shipped to the same address and this made them suspicious of fraud (eventually declining a range of such orders).

Ultimately, we addressed these problems by using a range of individual residential addresses (volunteered by researchers in our group). This approach significantly increased overhead for our group, however, and required that a large number of our members make regular trips to the post office to sign for international packages.

Receiving each individual shipment, inventorying its contents, shipping information and custom slip, and then mapping it back to an associated purchase transaction was a time consuming task and one fraught with ambiguity. The mapping challenge was greatly eased when we used EMS international shipping (a more expensive shipping option in which a tracking number is associated with an order and appears on the packaging) at the cost of roughly an additional $20 per order. Generally, items were packaged as advertised (including a proper customs declaration) but occasionally we received items where the goods had been secreted inside other items (e.g., handicrafts).

4.2.4 Consistency

With this ability to make purchases from underground vendors, we collected order numbers from ten different affiliate programs. While our initial observations of order number monotonicity are quite suggestive, we need to consider other possible explanations and confounding factors as well. Here we evaluate the data for *internal consistency*—the degree to which the data appears best explained by the *sequential update hypothesis* rather than other plausible explanations. At the end of the paper we also consider the issue of *external consistency* using "ground truth" revenue data for one program.

Sequential update

The fundamental premise underlying our purchase-pair technique is that order numbers increment sequentially for each attempted order. The monotone sequences that we observe accord with this hypothesis, but could arise from other mechanisms. Alternate interpretations include that updates are monotone but not sequential (e.g., incrementing the order number by a small, varying number for each order) or that order numbers are

derived from timestamps (i.e., that each order number is just a normalized representation of the time of purchase, and does not reflect the number of distinct purchase attempts).

To test these hypotheses, we executed back-to-back orders (i.e., within 5–10 seconds of one another) for each of the programs under study. We performed this measurement at least twice for all programs (excepting EvaPharmacy, which temporarily stopped operation during our study). For eight of the programs, every measurement pair produced a sequential increment. The GlavMed program also produced sequential increments, but we observed one measurement for which the order number incremented by two, likely simply due to an intervening order out of our control. Finally, we observed no sequential updates for Rx–Promotion even with repeated back-to-back purchase attempts. However, upon further examination of 35 purchases, we noticed that order numbers for this program are always odd; for whatever reason, the Rx–Promotion order processing system increments the order number *by two* for each order attempt. Adjusting for this deviation, our experiments find that on finer time scales, every affiliate program behaves consistently with the sequential update hypothesis.

We need however to consider an alternate hypothesis for this same behavior: that order numbers reflect normalized representations of timestamps, with each order implicitly serialized by the time at which it is received. This "clock" model does not appear plausible for fine-grained time scales. Our purchases made several seconds apart received sequential order numbers, which would require use of a clock that advances at a somewhat peculiar rate—slowly enough to risk separate orders receiving the same number and violating the uniqueness property.

A possible refinement to the clock model would be for a program to periodically allocate a block of order numbers to be used for the next T seconds (e.g., for $T = 3{,}600$), and after that time period elapses, advancing to the next available block. The use of such a hybrid approach would enable us to analyze purchasing activity over fine-grained time scales. But it would also tend towards misleading over-inflation of such activity on larger time scales, since we would be comparing values generated across gaps.

We test for whether the order numbers in our data fit with a clock model as follows. First, we consider the large-scale behavior of order numbers as seen across the different

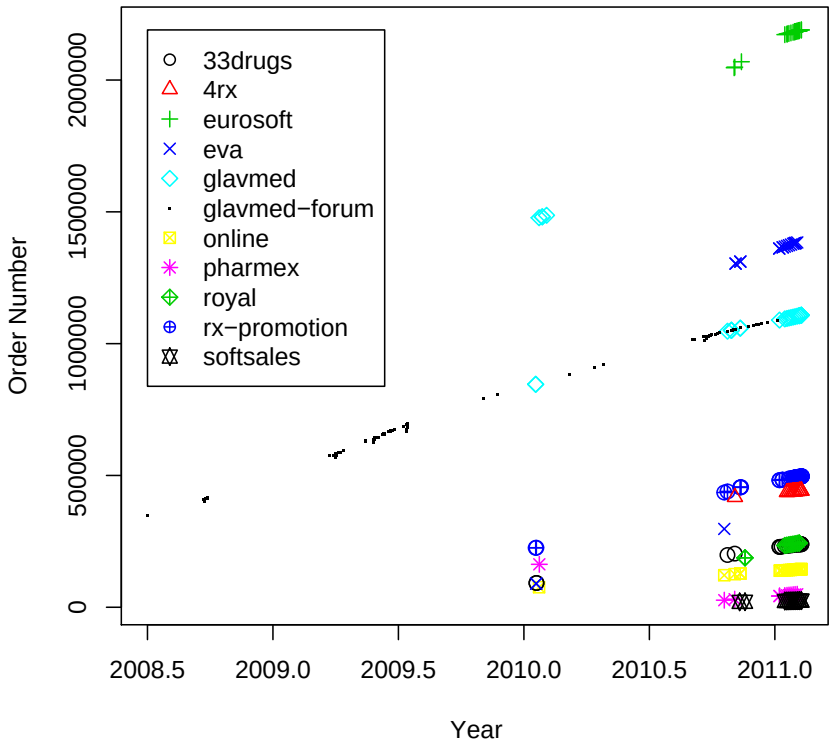


Figure 4.2: Order numbers (y-axis) associated with each affiliate program versus the time of attempted purchase (x-axis).

affiliate programs. Figure 4.2 plots for each program the order number associated with a purchase attempt made at a given time. We plot each of the 10 affiliate programs with a separate symbol (and varying shades, though we reuse a few for programs whose numbers are far apart). In addition, we plot with black points the order numbers revealed in the GlavMed discussion forum.

Three basic points stand out from the plot. First, all of the programs use order numbers distinct from the others. (We verified that neither of those closest together, 33drugs and Royal Software, nor Pharmacy Express and SoftSales, overlap.) Thus, it is not the case that separate affiliate programs share unified order processing.

Second, the programs nearly always exhibit monotonicity even across large time scales, ruling out the possibility that some programs occasionally reset their counters. (We discuss the outliers that manifest in the plot below.)

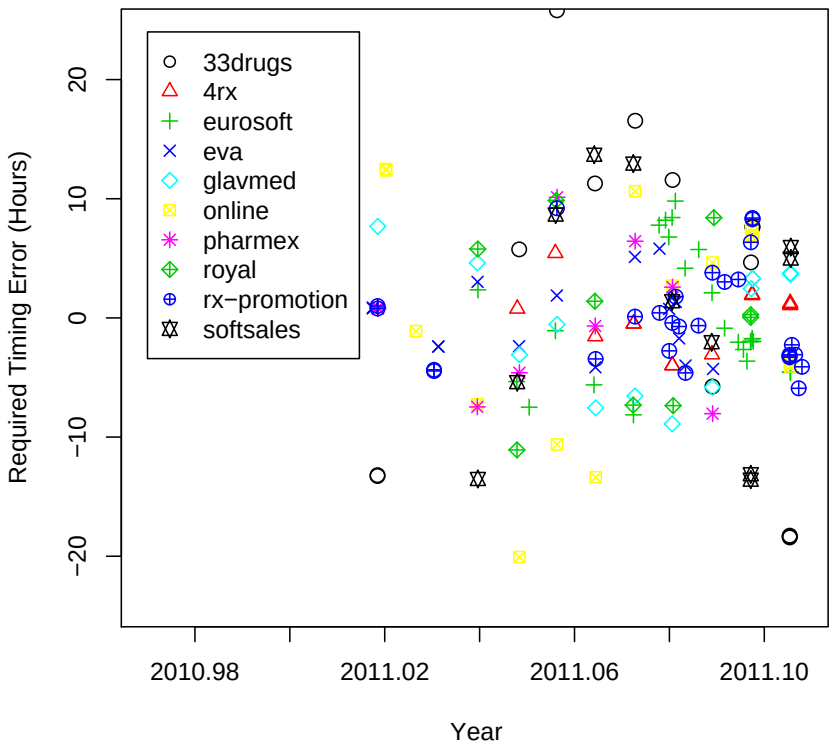


Figure 4.3: The amount of error—either in our measurement process, or due to batching of order numbers—required for each measurement in 2011 to be consistent with the Null Hypothesis that order numbers are derived from a clock that advances at some steady rate. Note that the y-axis is truncated at ± 24 hrs, though additional points lie outside this range.

Third, the GlavMed forum data is consistent with our own active purchases from GlavMed. In addition, the data for both has a clear downward concavity starting in 2009—inconsistent with use of clock-driven batches, but consistent with the sequential update hypothesis. Assuming that the data indeed reflects purchase activity, the downward concavity also indicates that the program has been losing customers, a finding consistent with mainstream news stories [39].

We lack such extensive data for the other programs, but can still assess their possible agreement with use of clock-driven batches, as follows. For each program, we consider the purchases made in 2011. We construct a least-squares linear fit between the order numbers of the purchases and the time at which we made them. If the order numbers

come from clock-driven batches (the Null Hypothesis), then we would expect that all of the points associated with our purchases to fall near the fitted line. Accordingly, for each point we compute how far we would have to move it along the x-axis so that it would coincide with the line for its program. If the Null Hypothesis is true, then this deviation in time reflects the *error* that must have arisen during our purchase measurement: either due to poor accuracy in our own time-keeping, or because of the granularity of the batches used by the program for generating order numbers.

Figure 4.3 plots this residual error for each affiliate program. For example, in the lower right we see a point for a 33drugs purchase made in early February 2011. If the Null Hypothesis holds, then the purchaser's order number reflects a value that should have appeared 18 hours earlier than when we observed it. That is, either we introduced an error of about 18 hours in recording the time of that purchase; or the program uses a batch-size of 18+ hours; or the Null Hypothesis fails to hold.

For all ten of the affiliate programs, we find many purchases that require timing errors of many hours to maintain consistency with the Null Hypothesis. (Note that we restrict the y-axis to the range ± 24 hr for legibility, although we find numerous points falling outside that range as well.) In addition, we do not discern any temporal patterns in the required errors, such as would be the case if the least-squares fit was perturbed by an outlier. Finally, if we extend the analysis out to November 2010 (not shown), we find that the required error *grows*, sometimes to 100s of hours, indicating that the discrepancy does not result from a large batch size such as $T = 1$ day.

Given this evidence, we reject the Null Hypothesis that the order numbers derive from a clock-driven mechanism. We do however find the data consistent with the sequential update hypothesis, and so proceed from this point on the presumption that indeed the order numbers grow sequentially with each new purchase attempt.

Payment independence

We placed most of our orders using cards underwritten by Visa. We selected Visa because it is the dominant payment method used by these affiliate programs (few accept MasterCard, and fewer still process American Express). However, it is conceivable that programs allocate distinct order number ranges for each distinct type of payment. If so,

then our Visa-based orders would only witness a subset of the order numbers, leading us to underestimate the total volume of purchase transactions. To test this question, we acquired several prepaid MasterCard cards and placed orders at those programs that accept MasterCard (doing so excludes Rx–Promotion, GlavMed, 4RX and Online Pharmacy). In each case, we found that Visa purchases made directly before and after a MasterCard purchase produced order numbers that precisely bracketed the MasterCard order numbers as well.

Outliers

Out of the 324 samples in our dataset, we found a small number of outliers (six) that we discuss here. Almost all come from the GlavMed program. The outliers fall into two categories: two singleton outliers completely outside the normal order number range for the program, and one group of four internally consistent order numbers that were slightly outside the expected range, violating monotonicity. We discuss these in more detail here, as well as their possible explanations.

The first singleton outlier was a purchase placed at a Web site that is clearly based on the SE2 engine built by GlavMed. However, the returned order number was close to 16000 when co-temporal orders from all other GlavMed sites returned orders closer to 1080000. The site differs in a number of key features, including a unique template not distributed in the standard package made available to GlavMed affiliates, a different support phone number, different product pricing, and purchases processed via a different acquiring bank than used by all other GlavMed purchases. Taken together, we believe this reflects a site that is simply using the SE2 engine, but is not in fact associated with the GlavMed operation.[7]

The second outlier occurred in a very early (January 2010) purchase from a Pharmacy Express affiliate, which returned an order number much higher than any seen in later purchases. We have no clear explanation for this incongruity, and other key structural and payment features match, but we note that the order numbers returned in all subsequent Pharmacy Express transactions are only five digits long, and that over nine

[7] We have found third parties contracting for custom GlavMed templates on popular "freelancer" sites, giving reason to believe that independent innovation exists around the SE2 engine created by GlavMed.

months pass between this initial outlier and all subsequent purchases. Consequently, we might reasonably explain the discrepancy by a decision to reset the order number space at some point between January and October.

Finally, we find a group of four early GlavMed purchases whose order numbers are roughly the same magnitude, but occur out of sequence (i.e., given the rate of growth seen in the other GlavMed order numbers, these four are from a batch that will only be used sometime in 2013). These all occurred together in the last two weeks of January 2010. This small outlier group remains a mystery, and suggests either that GlavMed might maintain a parallel order space for some affiliates, or that they reflect a "counterfeit" GlavMed operation. The remaining 21 GlavMed purchase samples, as well as the 122 opportunistically gathered order numbers (occurring both before and after January 2010), all use consistent order numbering.

While we cannot completely explain these few outliers, they represent less than 2% percent of our dataset. We also have found no unexplained instances within the last 12 months. We remove these six data points in the remainder of our analysis.

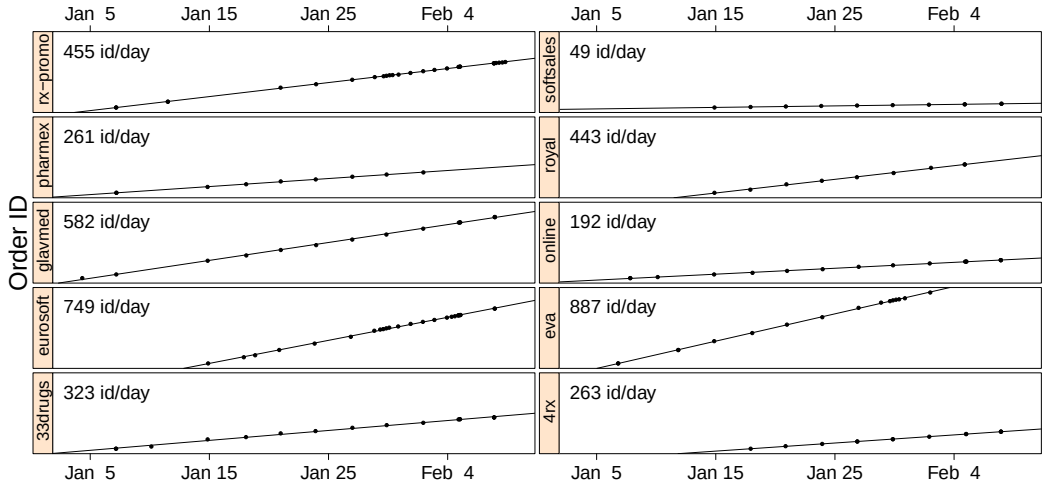


Figure 4.4: Collected data points and best fit slope showing the inferred order rate for ten different spam-advertised affiliate programs. Order numbers are zero-normalized and the vertical scale of each plot is identical.

4.2.5 Order rates

Under these assumptions, we can now estimate the rate of orders seen by each enterprise. Figure 4.4 plots the 2011 data points for each of the 10 programs. We also plot the least squares linear interpolation as well as the slope parameter of this line—corresponding to the number of orders received per day on average. During this time period, daily order rates for pharmacy programs vary from a low of 227 for Rx–Promotion (recall that their order IDs increment by two for each order) up to a high of 887 for EvaPharmacy (software programs range between 49 and 749). Together, these reflect a monthly volume of over 82,000 pharmaceutical orders and over 37,000 software orders. Again, these numbers reflect upper bounds on completed orders, since undoubtedly some fraction of these attempted orders are declined; however, it seems clear that order volume is substantial.

We also note that while order volume is quite consistent across January and February, there are significant fall offs for some programs when compared to the data gathered earlier. For example, during 2010, the average number of Rx–Promotion orders per day was 385, 70% greater than during the first two months of 2011. Similarly, 2011 GlavMed orders are off roughly 20% from their 2010 pace, and EvaPharmacy saw a similar decline as compared to October and November of that year. Other programs changed little and maintained a stable level of activity.

4.3 Purchasing behavior

While the previous analysis demonstrates that pharmaceutical affiliate programs are receiving a significant volume of orders, it reveals little about the source of these orders or their contents. In this section, we use an opportunistic analysis of found server log data to explore these issues for one such affiliate program.

4.3.1 EvaPharmacy image hosting

In particular, we examine EvaPharmacy, a "top 5" spam-advertised pharmacy affiliate program.[8] In monitoring EvaPharmacy sites we observed that roughly two thirds "outsourced" image hosting to compromised third-party servers (typically functioning Linux-based Web servers). This behavior was readily identifiable because visits to such sites produced HTML code in which each image load was redirected to another server—addressed via raw IP address—at port 8080.

We contacted the victim of one such infection and they were able to share IDS log data in support of this study. In particular, our dataset includes a log of HTTP request streams for a compromised image hosting server that was widely used by EvaPharmacy sites over five days in August of 2010. While the raw IP addresses in our dataset have been anonymized (consistently), they have first been geolocated (using MaxMind) and these geographic coordinates are available to us. Thus, we have city-level source identifiability as well as the contents of HTTP logs (including timestamp, object requested, and referrer).

Through repeated experimentation with live EvaPharmacy sites, we inferred that the site "engine" can use dynamic HTML rewriting (similar to Akamai) to rewrite embedded image links on a *per visit* basis. On a new visit (tracked via a cookie), the server selects a set of five compromised hosts and assigns these (apparently in a quasi-random fashion) to each embedded image link served. During the five-day period covering our log data, our crawler observed 31 distinct image servers in use. However, our particular server was apparently disproportionately popular, as it appears in 31% of all contemporaneous visits made by our URL crawler (perhaps due to its particularly good connectivity). In turn, each image server hosts an nginx Web proxy able to serve the entirety of the image corpus.

4.3.2 Basket inference

Since the log we use is limited to embedded Web page images, and in fact only includes one fifth of the images fetched during a particular visit, there are considerable challenges involved in inferring item selection purely from this data. We next discuss

[8] Our page classifiers [46] identified EvaPharmacy in over 8% of pharmacy sites found in spam-advertised URLs over three months, with affiliates driving traffic to over 11,000 distinct domains.

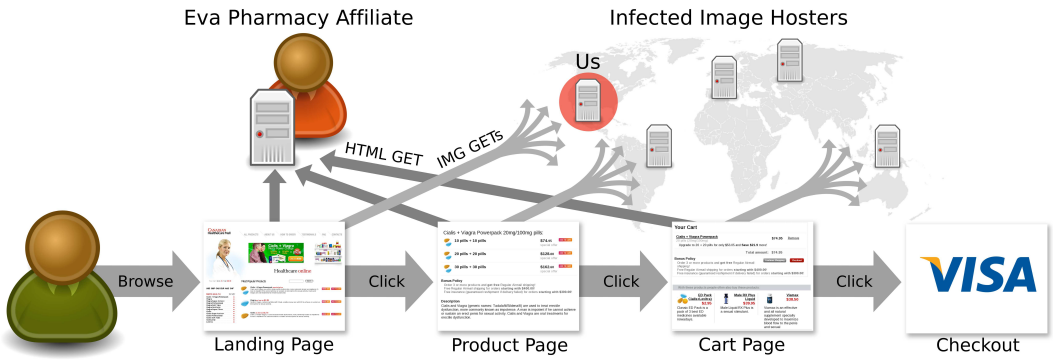


Figure 4.5: How a user interacts with an EvaPharmacy Web site, beginning with the landing page and then proceeding to a product page and the shopping cart. The main Web site contains embedded images hosted on separate compromised systems. When a browser visits such pages, the referrer information is sent to the image hosting servers for every new image visited.

how this inference technique works (illustrated at a high level in Figure 4.5) as well as its fundamental limitations.[9]

We mapped out the purchasing workflow involved in ordering from an EvaPharmacy site, and observed that all purchases involve visiting four key kinds of pages in order: landing, product, shopping cart, and checkout. The landing page generally includes over 40 distinct embedded images. Thus, even though images are split among five servers, it is highly likely that multiple objects from each landing page are fetched via our server (each with a referrer field identifying the landing page from which it was requested).[10] We observe 752,000 distinct IP addresses that visited and included referrer information during our five-day period.

When a visitor selects a particular drug from the landing page, the reply takes them to an associated product page. This page in turn prompts them to select the particular dosage and quantity they wish to purchase. The precise construction of product pages differs between the set of site templates (i.e., storefront brands) used by EvaPharmacy. However, all include at least a few new images not found on the landing page, and the most popular template fetches five additional images. The number of additional images

[9]This general approach is similar in character to Moore and Clayton's inference of phishing page visits from Webalizer logs [55].

[10]We validated this observation using our crawled data, which showed that the landing pages using :8080 image hosting always used five distinct servers. Thus, any image server assigned to a particular visit is guaranteed to see the landing page load for that visit.

varies on a per-template basis, not a per-product basis within each template. Thus, for some templates we may have less opportunity to observe what product the user selects, but this does not affect our estimate of the *distribution* of products selected, because the diminished opportunity is not correlated with particular products.

Next, upon selecting a product, the user is taken to the shopping cart page, which again includes a large number (often a dozen or more) of new images representing *product recommendations*. We observe 4,879 cart visits from 3,872 distinct IP addresses. This allows us to estimate a product-selection conversion rate: the fraction of visitors who select an item for purchase. Based on the total number of visitors where we have referrer information, the conversion percentage on an IP basis is 0.5%.[11] Of these, 3,089 cart additions have preceding visits to product pages, which allows us to infer the selected product. To quantify overall shopping cart addition activity, we compare the total number of visits to the number of visits to the shopping cart page. To quantify individual item popularity, we examine the subset of visits for which the customer workflow allows us to infer which specific item was added to the cart.

There are three key limitations to this approach. First and foremost, the final page in the purchasing workflow—the checkout page—generally does not include unique image content, and thus does not appear in our logs (even if it did, our approach could not determine whether checkout completed correctly). Thus, we can only observe that a user inserted an item into their cart, but not that they completed a purchase attempt. In general, this is only an issue to the degree that shopping cart abandonment correlates with variables of interest (e.g., drug choice). The second limitation is that pages typically use the same image for all dosages and quantities on a given product page, and therefore we cannot distinguish these features (e.g., we cannot distinguish between a user selecting 120 tablets of 25mg Viagra tablets vs. an order of 10 tablets, each of 100mg). Finally, we cannot disambiguate multiple items selected for purchase. When a user visits a product page followed by the shopping cart page, we can infer that they selected the associated product. However, if the visitor then *continues* shopping and visits additional product pages, we cannot determine whether they added these products or simply examined them (subsequent visits to the shopping cart page add few new recommended products;

[11] For comparison, the analysis in Chapter 3 measured a visit-to-product-selection conversion rate of 2%.

recommendations appear based on the first item in the cart). We choose the conservative approach and only consider the products that we are confident the user selected, which will cause us to under-represent those drugs typically purchased together.

Another issue is that pharmacy formularies, while largely similar, are not identical between programs. In particular, some pharmacy programs (e.g., Online Pharmacy) offer Schedule II drugs (e.g., Oxycodone and Vicodin). However, since EvaPharmacy does not sell such drugs, our data does not capture this category of demand.

Finally, our dataset also has potential bias due to the particular means used to drive traffic to it. We found that 45 of the 50 top landing pages observed in the hosting data also appeared in our spam-driven crawler data, demonstrating directly that these landing pages were advertised through email spam. While these pages could also be advertised using less risky methods such as SEO, this seems unlikely since spam-advertised URLs are swiftly blacklisted [41]. Thus, we suspect (but cannot prove) that our data may *only* capture the purchasing behavior for the spam-advertised pharmacies; different advertising vectors could conceivably attract different demographics with different purchasing patterns.

Given these limitations, we now report the results of two analyses: product popularity (what customers buy) and customer distribution (where the money comes from).

4.3.3 Product popularity

Our first analysis focuses on simple popularity: what individual items users put into their shopping carts (Table 4.3a) and what broad (seller-defined) categories of pharmaceuticals were popular (Table 4.3b) during our measurement period. Although naturally dominated by the various ED and sexually-related pharmaceuticals, we find a surprisingly long tail; indeed, 38% of all items added to the cart were not in this category. We observed 289 distinct products, including popular mass-market products such as Zithromax (31), Acomplia (27), Nexium (26), and Propecia (27); but also Cipro (11; a commonly prescribed antibiotic), Actos (6; a treatment for Type 2 diabetes), Buspar (12; anti-anxiety), Seoquel (9; anti-schitzophrenia), Clomid (8; ovulation inducer), and Gleevec (1; used to treat Leukemia and other cancers).

This in turn explains why such online pharmacies maintain a comprehensive in-

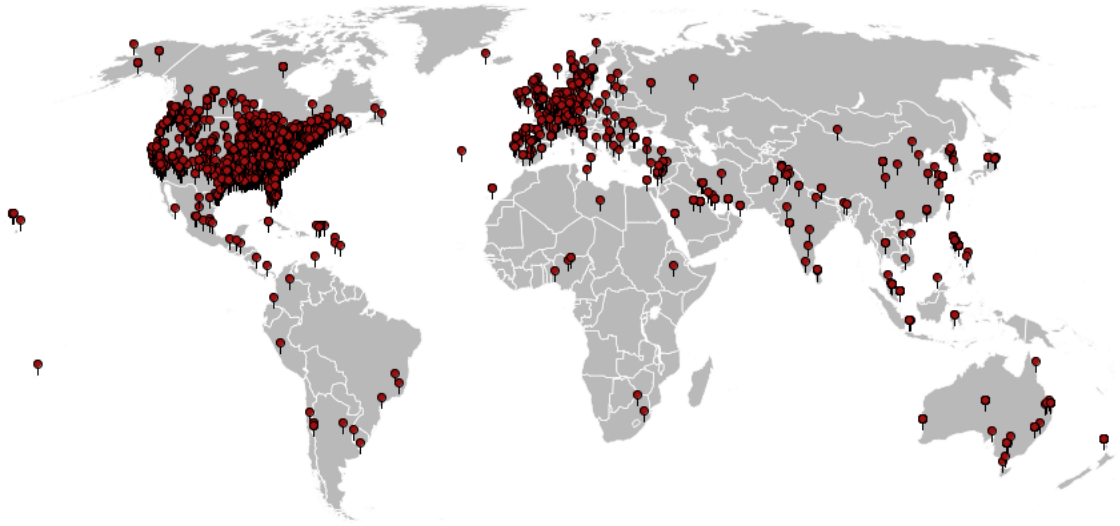


Figure 4.6: The geographic distribution of those who added an item to their shopping cart.

ventory: not only does a full formulary lend legitimacy, but it also represents a significant source of potential revenue.

We also comprehensively crawled an EvaPharmacy site for pricing data and calculated the *minimum* estimated revenue per purchase (also shown for the top 18 products in Table 4.3a). Combining this data with our measurement of item popularity, we calculate a minimum weighted-average item cost of $76 plus $15 for shipping and handling. This weighted average assumes visitors always select the minimum-priced item for any given purchase, and that the final purchases have the same distribution as for items added to the user's shopping cart.

4.3.4 Customer distribution

We next examine the geographic component of the EvaPharmacy customer base. Figure 4.6 shows the geolocated origin for all shopping cart additions. We observe that EvaPharmacy has a vast advertising reach, producing site visits from 229 distinct countries or territories. However, this reach is not necessarily all that useful: the population *actively engaging* with EvaPharmacy sites and placing orders is considerably less diverse than the superset simply visiting (perhaps inadvertently or due to curiosity). For example, the Philippines constitutes 4% of the visitors, but only 1% of the additions to the shopping

Table 4.2: The top 15 countries and the percentage of visitors who added an item to their shopping cart.

Country	Visits	Cart Additions	Added Product
United States	517,793	3,707	0.72%
Canada	50,234	218	0.43%
Philippines	42,441	39	0.09%
United Kingdom	39,087	131	0.34%
Spain	26,968	59	0.22%
Malaysia	26,661	31	0.12%
France	18,541	37	0.20%
Germany	15,726	56	0.36%
Australia	15,101	86	0.57%
India	10,835	17	0.16%
China	8,924	30	0.34%
Netherlands	8,363	21	0.25%
Saudi Arabia	8,266	36	0.44%
Mexico	7,775	17	0.22%
Singapore	7,586	17	0.22%

cart. Overall, countries other than the U.S., Canada, and Western Europe generate 29% of the visitors but only 13% of the items added to the shopping cart. Conversely, the vast majority of shopping cart insertions originate from the U.S. and Canada (80%) or Europe (6%), reinforcing the widely held belief that spam-advertised pharmaceuticals are ultimately funded with Western Dollars and Euros.

The United States dominates both visits (54%) and cart additions (76%), and moreover has the highest rate of conversion between visit and shopping cart insertion (0.72%). Table 4.2 well illustrates this, listing the activity from the countries originating the most visits. This observation reinforces the conclusion that non-Western audiences offer ineffective targets for such advertising.

Finally, we also notice significant differences between the drug selection habits of Americans compared to customers from Canada and Western Europe. In particular, we divide the EvaPharmacy formulary into two broad categories: lifestyle drugs (defined as drugs commonly used recreationally, including “male-enhancement” items plus Human Growth Hormone, Soma and Tramadol) and non-lifestyle (all others, including birth

Table 4.3: Table (a) shows the top 18 product items added to visitor shopping carts (representing 66% of all items added). Table (b) shows the top 18 seller-defined product categories (representing 99% of all items).

(a)

Product	Quantity	Min order
Generic Viagra	568	$78.80
Cialis	286	$78.00
Cialis/Viagra Combo Pack	172	$74.95
Viagra Super Active+	121	$134.80
Female (pink) Viagra	119	$44.00
Human Growth Hormone	104	$83.95
Soma (Carisoprodol)	99	$94.80
Viagra Professional	87	$139.80
Levitra	83	$100.80
Viagra Super Force	81	$88.80
Cialis Super Active+	72	$172.80
Amoxicillin	47	$35.40
Lipitor	38	$14.40
Ultram	38	$45.60
Tramadol	36	$82.80
Prozac	35	$19.50
Cialis Professional	33	$176.00
Retin A	31	$47.85

(b)

Category	Quantity
Men's Health	1760
Pain Relief	232
Women's Health	183
General Hearth	135
Antibiotics	134
Antidepressants	95
Weight Loss	92
Allergy & Asthma	85
Heart & Blood Pressure	72
Skin Care	54
Stomach	41
Mental Health & Epilepsy	33
Anxiety & Sleep Aids	33
Diabetes	22
Smoking Cessation	22
Vitamins and Herbal Suppliments	18
Eye Care	15
Anti-Viral	14

control pills). We find that while U.S. customers select non-lifestyle items 33% of the time, Canadian and Western-European customer selections concentrate far more in the lifestyle category—only 8% of all items placed in a shopping cart are non-lifestyle items. We surmise that this discrepancy may arise due to differences in health care regimes; drugs easily justified to a physician may be fully covered under state health plans in Canada and Western Europe, leaving an external market only for lifestyle products. Conversely, a subset of uninsured or under-insured customers in the U.S. may view spam-advertised, no-prescription-required pharmacies as a competitive market for meeting their medical needs. To further underscore this point, we observe that 85% of *all* non-lifestyle drugs are selected by U.S. visitors.

4.4 Revenue estimation

Combining the results from estimates on the order rate per program and estimates of the shopping cart makeup, we now estimate total revenue on a per-program basis.

4.4.1 Average price per order

The revenue model underlying our analysis is simple: we multiply the estimated order rate by the average price per order to arrive at a total revenue figure over a given unit of time. However, we do not know, on a per-program basis, the actual average purchase price. Thus, we explore three different approximations, all of which we believe are conservative.

First, for online pharmacies we use the static value of roughly $100 that we used for the analysis in Chapter 3. This study only considered one particular site, covered only 28 customers, and was unable to handle more than a single item placed in a cart (i.e., it could not capture information about customers buying multiple items).

We also consider a second approximation based on the minimum priced item (including shipping) on the site for each program under study. Since sites can have enormous catalogs, we restrict the set of items under consideration as follows. For pharmacy sites, we consider the top 18 most popular items as determined by the analysis of EvaPharmacy in Section 4.3 (these top 18 items constituted 66% of order volume in our analysis). For each of these items present in the target pharmacy, we find the minimum-priced instance (i.e., lowest dosage and quantity) and use the overall minimum as our per-order price. For small deviations between pharmacy formularies (e.g., different Viagra store-brand variants) we simply substitute one item for the other. We repeat this same process for software, but since we do not have a reference set of most popular items for this market, we simply use the declared "bestsellers" at each site (16 at Royal Software, 36 and SoftSales and 76 at EuroSoft)—again using the minimum priced item to represent the average price per order.

Finally, we calculate a "basket-weighted average" price using measured popularity data. For pharmacies we again consider the 18 most popular EvaPharmacy items and extract the overlap set with other pharmacies. Using the relative frequency of elements in this intersection, we calculate a popularity vector that we then use to weight the minimum item price; we use the sum of these weights as the average price per order. Intuitively, this approach tries to accommodate the fact that product's have non-uniform popularity, while still using the conservative assumption that users order the minimum dosage and quantity for each item. Note that we implicitly assume that the distribution of drug popularity

Table 4.4: Estimated monthly order volume, average purchase price, and monthly revenue (in dollars) per affiliate program using three different per-order price approximations.

Affiliate Program	orders/mo	Chapter 3		Min product price		Basket-weighted average	
		single order	rev/month	single order	rev/month	single order	rev/month
33drugs	9,862	$100	$980,000	$45.00	$440,000	$57.25	$560,000
4RX	8,001	$100	$800,000	$34.50	$280,000	$95.00	$760,000
EuroSoft	22,776	N/A	N/A	$26.50	$600,000	$84.50	$1,900,000
EvaPharmacy	26,962	$100	$2,700,000	$50.50	$1,300,000	$90.00	$2,400,000
GlavMed	17,933	$100	$1,800,000	$54.00	$970,000	$57.00	$1,000,000
Online Pharmacy	5,856	$100	$590,000	$37.00	$220,000	$58.00	$340,000
Pharmacy Express	7,933	$100	$790,000	$51.00	$410,000	$58.75	$460,000
Royal Software	13,483	N/A	N/A	$55.25	$750,000	$133.75	$1,800,000
Rx–Promotion	6,924	$100	$690,000	$45.00	$310,000	$57.25	$400,000
SoftSales	1,491	N/A	N/A	$20.00	$30,000	$134.50	$200,000

holds roughly the same between online pharmacies.[12]

We repeated this analysis, as before, with site-declared best-selling software packages. To gauge relative popularity, we searched a large BitTorrent metasearch engine (isohunt.com), which indexes 541 sites tracking over 6.5 million torrents. We assigned a popularity to each software item in proportion to the sum of the seeders and leechers on all torrents matching a given product name. We then weighted the total prices (inclusive of any handling charge) by this popularity metric to arrive at an estimate of the average order price.

4.4.2 Revenue

Finally, to place a rough estimate on revenue, we multiply the 2011 order volume measurements shown in Figure 4.4 against each of the previously mentioned approximations, summarized in Table 4.4. In general, the approximation from Chapter 3 is the largest, followed by basket-weighted average and then minimum product price. However, for pharmaceutical programs the difference between product prices is not large, and thus the minimum and basket-weighted estimates all lie within 2X of one another. Software programs see much more variation in price, and hence the difference between the minimum and basket-weighted revenue estimates can be substantial.

Using the basket-weighted approximation, we find that both GlavMed and Eva-

[12] One data point supporting this view is Rx–Promotion's rank-ordered list of best selling drugs. The ten most popular items sold by both pharmacies are virtually the same and ranked in the same order.

Pharmacy produce revenues in excess of $1M per month, with all but two over $400K. Surprisingly, software sales also produce high revenue—less due to high prices than high order volumes. It remains for future work how to further validate how closely order volumes track successfully completed *orders* for this market niche.

4.4.3 External consistency

While we put considerable care into producing these estimates, a number of biases remain unavoidable. First, while our order volume data has internal consistency (and consistency with order number implementations in common shopping cart software), we could not capture the impact of order declines. Thus, we have a somewhat optimistic revenue estimate, since surely some fraction of orders will not complete.

On the other hand, our estimates of average order revenue are themselves conservative in several key ways. First, they assume that all purchasers select only a single item. Second, they assume that when purchasing an item, all users select the minimum dosage and quantity. Finally, for pharmaceuticals we need to keep in mind that EvaPharmacy does not carry "harder" drugs found at other sites, such as Schedule II opiates. We have found anecdotal evidence that these drugs are highly popular at such sites, but our methodology does not offer any means to consider their impact. Such items are also typically more expensive than other drugs (e.g., the cheapest Hydrocodone order possible at one popular pharmacy is $186 plus shipping). Thus, this other factor will cause us to *underestimate* the true revenue per order.

Our intuition is that such factors are modest, and our estimates capture—within perhaps a small constant factor—the true level of financial activity within each enterprise. However, absent ground truth data for program revenues, it is not generally possible to validate our model and hence verify that our measurements actually capture reality. In general, this kind of validation is rarely possible since the actors involved are not public companies and do not make revenue statements available.

Due to an unusual situation, however, we were able to acquire such information for one program, Rx–Promotion. In particular, a third party made public a variety of information, including multiple months of accounting data, for Rx–Promotion's payment

processor.[13] While we cannot validate the provenance of this data, its volume and specificity make complete fabrication unlikely. In addition, given that our research covers only a small subset of this data, it seems further unlikely that any fabrication would closely match our own independent measurements.

Unfortunately, we do not have payment ledgers precisely covering our 2011 measurement period. Instead, we compare against a similar period six months earlier for which we do have ground truth documentation, 27 consecutive days from the end of Spring, 2010. These two periods are comparable because during both times Rx–Promotion had significant difficulty processing orders on "controlled" drugs (indeed, during the 2011 period such drugs had been removed from the standard formulary on Rx–Promotion affiliates).[14]

Based on this data, we find that between May 31 and June 26, 2010, Rx–Promotion's turnover via electronic payments was $609K.[15] Using our estimate of 385 orders per day in 2010 (see Section 4.2), this is consistent with an average revenue per order of $58, very similar to our basket-weighted average order price estimate of $57. While we suspect that both estimates are likely off (with the number of true June 2010 orders likely less due to declines, and January 2011 price-per-order likely higher due to conservatism in our approximation), they are sufficiently close to one another to support our claim that this approach can provide a rough, but well-founded estimate (i.e., within a small constant factor) of program revenue.

4.5 Summary

When asked why he robbed banks, Willie Sutton famously responded, "Because that's where the money is." The same premise is frequently used to explain the plethora

[13] While our legal advisers believe that the prior public disclosure of this data allows its use in a research context, we chose not to unnecessarily antagonize the payment services provider by naming them here.

[14] During periods when such drugs were sold *en masse*, the overall Rx–Promotion revenue was frequently doubled.

[15] Interestingly, this data also provides useful information about refunds and chargebacks (together about 10% of revenue) as well as processing fees (roughly 8.5%). Thus, the gross revenue delivered to Rx–Promotion in June 2010 was likely closer to $489K. Finally, since roughly 40% of successful order income is paid to affiliates on a commission basis, that leaves only $270K (44% of gross) for fulfillment, administrative costs, and profit.

of unwanted spam that fills our inboxes, pollutes our search results and infests our social networks—spammers spam because they can make money at it. However, a key question has long been how much money, and from whom? In this chapter we provide what we believe represents the most comprehensive attempt to answer these questions to date. We have developed new inference techniques: one to estimate the rate of new orders received by the very enterprises whose revenue drives spam, and the other to characterize the products and customers who provide that same revenue. We provide quantitative evidence showing that spam is ultimately supported by Western purchases, with a particularly central role played by U.S. customers. We also provide the first sense of market size, with well over 100,000 monthly orders placed in our dataset alone. Finally, we provide rough but well-founded estimates of per-program revenue. Our results suggest that while the spam-advertised pharmacy market is substantial, with annual revenue in the many tens of millions of dollars, it has nowhere near the size claimed by some, and indeed falls vastly short of the annual expenditures on technical anti-spam solutions.

This chapter characterizes both the demand for spam-advertised goods and the suppliers that meet this demand. Both the weaknesses of the spammers' compromised machine infrastructure and opportunistic data collection have enabled this research. These data collection methodologies have allowed us to analyze the magnitude of several spamming operations at once and take a deep look at the worldwide demand for spam-advertised pharmaceuticals that supports this market.

Chapter 4, in part, is a reprint of the material as it appears in Proceedings of USENIX Security 2011. Kanich, Chris; Weaver, Nicholas; McCoy, Damon; Halvorson, Tristan; Kreibich, Christian; Levchenko, Kirill; Paxson, Vern; Voelker, Geoffrey M., Savage, Stefan. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, is a reprint of the material as it appears in Proceedings of USENIX CSET 2011. Kanich, Chris; Chachra, Neha; McCoy, Damon; Grier, Chris; Wang, David; Motoyama, Marti; Levchenko, Kirill; Savage, Stefan; Voelker, Geoffrey M. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Conclusion

When conducting computer security research, considering the motivations of the adversary can provide precious insight into new approaches to the problem at hand. The immense success of the Internet as a medium for communication and commerce has attracted many actors who abuse the system for financial gain. While understanding the technical aspects of these perpetrators' schemes is necessary to implementing defenses, understanding how they achieve their financial gains is crucial to developing countermeasures that not only prevent their current abuses, but target their bottom line as a lasting solution to the problem.

To develop an understanding of Internet scams as business processes, one must determine what questions to ask, and how to procure the data; the finances of Internet scammers are not open to public perusal or analysis. A fine first task is to determine how much money they make, and who purchases their goods. Before this work, our understanding of these variables was limited; without sound estimates, our characterization of the adversary is rough or even incorrect.

This dissertation posits that one can understand the business processes of Internet scammers through infrastructure infiltration, providing important insights into the symptoms of abuse on the Internet. Through interposing on the Storm botnet, analyzing the image load logs of a compromised image host, and the purchase pair methodology, we have shown that we can analyze and understand the business processes of several different underground actors, learning important facts about their revenue streams, customers, and distribution methods.

5.1 Impact

This research gives a better understanding of the root cause of spam—that its perpetrators can make a healthy profit—allowing defenders to look beyond the classification problem to a wider array of defensive possibilities. Following the money allowed us to discover that 95% of all spam received at multiple spam traps over three months shares just three banks for their credit card processing infrastructure [46]. In addition, analyzing the difficulty of different circumvention strategies for the spammers helps us determine where to best focus our defensive efforts. Not only is there a great concentration of spammers at just a few banks, but finding new payment processors is far more difficult than finding new bots to send the spam, contracting with new hosting providers to host the Web storefronts, or replacing other technical aspects of their operation. This work was enabled by our continued investigation of the spam economy and our understanding of the structure of the business that we gained investigating the Storm botnet.

Our experience infiltrating spamming botnets, for example, led to the Botnet Judo system [59]. Botnet Judo capitalizes on our understanding of botnet-based spam distribution to build highly effective filters tailored to the spamming templates used by several individual botnets. By running spamming bots in a contained environment with no outbound spamming connectivity, we redirect the spam output to the Judo system and infer the templates for messages very soon after they are disseminated into the botnet. This template inference allows us to create regular expressions that very specifically describe the spam output created by individual templates. As universal spam defenses, this approach can only filter a small percentage of all spam, but as these detectors are specific to individual templates, they are very unlikely to cause false positives by filtering desired email. The Botnet Judo system arose both out of our expertise running spamming bots in captivity and our insights into the potential profits of botnet-based spam distribution.

5.2 Future Directions

The adversarial nature of security coupled with the ever-present profit motive ensures that new scams will present themselves and necessitate informed creation of effective defenses. Expertise gained through this data-driven approach to security can be

applied throughout the ecosystem to current and future threats alike. New online scams are discovered every day; characterizing the demand for and revenue from these scams is an important step in building effective countermeasures and in turn creating a safer Internet.

With the methodology of botnet infiltration in mind, one can use a similar strategy to characterize the successes of another class of online criminal: those who steal online financial credentials. While sending spam to advertise goods is one revenue scheme, a more direct but riskier approach is simply to steal the target's money, usually by stealing the online financial credentials from malware-infected victims. The technical capabilities necessary for malware and phishing-based compromises are well understood, but what happens after credentials are stolen is less clear. Through data-sharing agreements with online financial companies, we can investigate the ecosystem of account credential theft and systematically map out the value chain employed by the criminals to monetize these accounts.

Our infiltration technique is not limited purely to technical approaches: one can also infiltrate the *social* networks of the adversary. As underground actors specialize, they must communicate and interface with each other to achieve their goals. We can capitalize on their need to trust and interact with others and participate in the markets to learn about their abusive operations. Not only will learning about the products, prices, and actors in these markets give us insight into the myriad scams run by bad actors on the Internet, but one might also learn valuable information about these scams by purchasing and testing the goods themselves. These research directions all utilize understanding and methodologies developed during the research comprising this dissertation, and, hopefully, will also be useful in better understanding the motivations and business processes of Internet scammers in different domains.

5.3 Final Thoughts

In this research we have capitalized on the side effects of the moral flexibility of the perpetrators of spam: the use of compromised hosts is an economic advantage for spammers, but is a fundamental weakness that we have exploited multiple times to

collect information regarding the inner workings of their operations. If scammers think twice about using compromised hosts in the future, this in itself is a positive result for Internet users; what is more likely is that scammers on the Internet continue to utilize compromised machines in their schemes. If this is indeed the case, we hope that we will still be able to capitalize on this weakness of their business model and better understand their operations through new infiltrations.

Bibliography

[1] C. Akass. Storm worm 'making millions a day'. http://www.computeractive.co.uk/pcw/news/1923144/storm-worm-millions-day, February 2008.

[2] D. S. Anderson, C. Fleizach, S. Savage, and G. M. Voelker. Spamscatter: Characterizing Internet Scam Hosting Infrastructure. In *Proc. of 16th USENIX Security*, 2007.

[3] J. Angwin. Elusive Spammer Sends EarthLink on Long Chase. http://online.wsj.com/article_email/SB105225593382372600.html, 2003.

[4] J. Aycock and N. Friess. Spam Zombies from Outer Space. Technical report, Department of Computer Science, University of Calgary, 2006.

[5] A. Back. Hashcash - A Denial of Service Counter-Measure. http://www.hashcash.org/papers/hashcash.pdf, 2002.

[6] J. Blosser and D. Josephsen. Scalable Centralized Bayesian Spam Mitigation with Bogofilter. In *Proc. of 18th USENIX LISA*, 2004.

[7] R. Boehme and T. Ho. The Effect of Stock Spam on Financial Markets. In *Proc. of 5th WEIS*, 2006.

[8] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. *OpenPGP Message Format*. IETF, 2007.

[9] CBL. Composite Blocking List. http://cbl.abuseat.org, 2003.

[10] D. Cook, J. Hartnett, K. Manderson, and J. Scanlan. Catching Spam Before it Arrives: Domain Specific Dynamic Blacklists. In *Proc. of 4th AISW-NetSec*, 2006.

[11] M. de Vivo, E. Carrasco, G. Isern, and G. de Vivo. A Review of Port Scanning Techniques. *Computer Communication Review*, 29(2):41–48, 1999.

[12] Direct Marketing Association. DMA Releases 5th Annual 'Response Rate Trends Report'. http://www.the-dma.org/cgi/disppressrelease?article=1008, 2007.

[13] C. Dwork and M. Naor. Pricing via Processing or Combatting Junk Mail. In *Proc. of 12th IACR CRYPTO*, 1993.

[14] S. E. Fahlman. Selling interrupt rights: A way to control unwanted e-mail and telephone calls. *IBM Systems Journal*, 41(4):759–766, 2002.

[15] C. Fleizach, G. M. Voelker, and S. Savage. Slicing Spam with Occam's Razor. In *Proc. 4th CEAS*, 2007.

[16] Forrester Data. Consumer Attitudes Toward Spam in Six Countries. http://www.bsacybersafety.com/files/Forrester_Consumer_Spam.pdf, 2004.

[17] L. Frieder and J. Zittrain. Spam Works: Evidence from Stock Touts and Corresponding Market Activity. *Berkman Center Research Publication*, 2006.

[18] P. Gburzynski and J. Maitan. Fighting the Spam Wars: A Remailer Approach with Restrictive Aliasing. *ACM Transactions on Interet Technology*, 4(1):1–30, 2004.

[19] L. H. Gomes, C. Cazita, J. M. Almeida, V. Almeida, and W. M. Jr. Characterizing a Spam Traffic. In *Proc. of 4th ACM IMC*, 2004.

[20] J. Goodman and R. Rounthwaite. Stopping Outgoing Spam. *Proc. of 5th ACM EC*, 2004.

[21] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: The Underground on 140 Characters or Less. In *Proc. of 17th ACM CCS*, New York, NY, USA, 2010.

[22] R. J. Hall. Channels: Avoiding unwanted electronic mail. In *Proc. of 1996 DIMACS Symposium on Network Threats*, 1996.

[23] M. Hanke and F. Hauser. On the Effects of Stock Spam E-mails. *Journal of Financial Markets*, 11(1):57–83, 2008.

[24] C. Herley and D. Florêncio. A Profitless Endeavor: Phishing as Tragedy of the Commons. In *Proc. of the 11th NSPW*, 2008.

[25] C. Herley and D. Florêncio. Economics and the Underground Economy. Black Hat, 2009.

[26] C. Herley and D. Florêncio. Nobody Sells Gold for the Price of Silver: Dishonesty, Uncertainty and the Underground Economy. In *Proc. of 8th WEIS*, 2010.

[27] J. Ioannidis. Fighting Spam by Encapsulating Policy in Email Addresses, 2003.

[28] Ipsos Public Affairs. Key Findings of the 2010 MAAWG Email Security Awareness and Usage Survey. http://www.maawg.org/system/files/2010_MAAWG-Consumer_Survey_Key_Findings.pdf, 2010.

[29] Ironport. 2008 Internet Security Trends. http://www.ironport.com/securitytrends/, 2008.

[30] P. Judge, D. Alperovitch, and W. Yang. Understanding and Reversing the Profit Model of Spam, 2005.

[31] J. Jung and E. Sit. An Empirical Study of Spam Traffic and the Use of DNS Black Lists. In *Proc. of 4th ACM IMC*, 2004.

[32] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. Spamalytics: An Empirical Analysis of Spam Marketing Conversion. In *Proc. of 15th ACM CCS*, 2008.

[33] C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, and S. Savage. The Heisenbot Uncertainty Problem: Challenges in Separating Bots from Chaff. In *Proc. of 1st USENIX LEET*, 2008.

[34] S. Karam. Cybercrime is more effective than drug trading. http://www.crime-research.org/news/29.11.2005/1666/, 2005.

[35] D. W. K. Khong. An Economic Analysis of Spam Law. *Erasmus Law and Economics Review*, 1(1):23–45, 2004.

[36] J. Kirk. Former spammer: 'I know I'm going to hell'. http://www.macworld.com/article/58997/2007/07/spammer.html, July 2007.

[37] D. Koblas and M. R. Koblas. SOCKS. In *Proc. of 3rd USENIX Security* , 1992.

[38] Kommersant. Spamming may become criminal offense. http://en.rian.ru/papers/20101202/161593138.html, 2010.

[39] B. Krebs. Spam Affiliate Program Spamit.com to Close. http://krebsonsecurity.com/2010/09/spam-affialite-program-spamit-com-to-close/, 2010.

[40] C. Kreibich, C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. On the Spam Campaign Trail. In *First USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'08)*, April 2008.

[41] C. Kreibich, C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage. Spamcraft: An Inside Look at Spam Campaign Orchestration. In *Proc. of 1st USENIX LEET*, 2009.

[42] C. Kreibich, N. Weaver, C. Kanich, W. Cui, and V. Paxson. GQ: Practical Containment for Measuring Modern Malware Systems . In *Proc. of 11th ACM IMC*, 2011.

[43] B. Laurie and R. Clayton. "Proof-of-work" Proves Not to Work. In *Proc. of 3rd WEIS*, 2004.

[44] LegitScript. Industry Trends: EvaPharmacy, 33Drugs (DrugRevenue) emerge as major Internet threats. http://legitscriptblog.com/2009/10/industry-trends-evapharmacy-33drugs-drugrevenue-emerge-as-major-internet-threats/, 2009.

[45] B. Leiba and J. Fenton. DomainKeys Identified Mail (DKIM): Using Digital Signatures for Domain Verification. In *Proc. of 4th CEAS*, 2007.

[46] K. Levchenko, A. Pitsillidis, N. Chachra, B. Enright, M. Félegyházi, C. Grier, T. Halvorson, C. Kanich, C. Kreibich, H. Liu, D. McCoy, N. Weaver, V. Paxson, G. M. Voelker, and S. Savage. Click Trajectories: End-to-End Analysis of the Spam Value Chain. In *Proc. of 32nd IEEE Symposium on Security and Privacy*, 2011.

[47] J. Levine. *DNS Blacklists and Whitelists*. IETF, 2010.

[48] T. Loder, M. Van Alstyne, and R. Wash. An Economic Answer to Unsolicited Communication. In *Proc of 5th ACM EC*, 2004.

[49] Magento. Magento Ecommerce Platform. http://www.magentocommerce.com/.

[50] S. Malinin. Spammers earn millions and cause damages of billions. http://english.pravda.ru/russia/economics/15-09-2005/8908-spam-0/, 2005.

[51] U. Manber. Finding Similar Files in a Large File System. In *Proc. of USENIX Winter 1994 Technical Conference*, 1994.

[52] Marshal. Sex, Drugs and Software Lead Spam Purchase Growth. http://www.m86security.com/newsitem.asp?article=748, 2008.

[53] J. Mastaler. Tagged Message Delivery Agent. http://tmda.net, 2001.

[54] P. Maymounkov and D. Mazières. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Proc. of 1st IPTPS*, 2002.

[55] T. Moore and R. Clayton. An Empirical Analysis of the Current State of Phishing Attack and Defence. In *Proc. of 6th WEIS*, 2007.

[56] J. Myers. *SMTP Service Extension for Authentication*. IETF, 1999.

[57] Y. Niu, Y. min Wang, H. Chen, M. Ma, and F. Hsu. A Quantitative Study of Forum Spamming Using Context-based Analysis. In *Proc. of 14th NDSS*, 2007.

[58] Paul Graham. So Far, So Good. http://paulgraham.com/sofar.html, 2003.

[59] A. Pitsillidis, K. Levchenko, C. Kreibich, C. Kanich, G. M. Voelker, V. Paxson, N. Weaver, and S. Savage. Botnet Judo: Fighting Spam with Itself. In *Proceedings of the 17th Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, Feb. 2010.

[60] J. B. Postel. *Simple Mail Transfer Protocol*. IETF, 1982.

[61] V. V. Prakash. Vipul's Razor. http://razor.sourceforge.net, 1999.

[62] A. Ramachandran, D. Dagon, and N. Feamster. Can DNSBLs Keep Up with Bots? In *Proc. of 3rd CEAS*, 2006.

[63] A. Ramachandran and N. Feamster. Understanding the Network-Level Behavior of Spammers. In *Proc. of ACM SIGCOMM*, 2006.

[64] A. Ramachandran, N. Feamster, and D. Dagon. Revealing Botnet Membership using DNSBL Counter-Intelligence. In *Proc. of 2nd USENIX SRUTI*, 2006.

[65] A. Ramachandran, N. Feamster, and S. Vempala. Filtering Spam with Behavioral Blacklisting. In *Proc. of 14th ACM CCS*, 2007.

[66] Rhyolite Corporation. Distributed Checksum Clearinghouse. http://www.rhyolite.com/dcc/, 2000.

[67] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. of 3rd Middleware*, 2001.

[68] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A Bayesian Approach to Filtering Junk E-Mail. In *Proc. of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[69] D. Samosseiko. The Partnerka - What is it, and why should you care? In *Proc. of Virus Bulletin Conference*, 2009.

[70] Senate Committee on Commerce, Science, and Transportation. Cybersecurity - Assessing Our Vulnerabilities and Developing An Effective Defense, 2009.

[71] A. Serjantov and R. Clayton. Modeling Incentives for Email Blocking Strategies. *Proc. of 4th WEIS*, 2005.

[72] Spamhaus. The policy block list. http://www.spamhaus.org/pbl/, 2012.

[73] Spamhaus. The spamhaus blocklist. http://www.spamhaus.org/sbl/, 2012.

[74] Spammer-X and S. Sjouwerman. *Inside The SPAM Cartel: Trade Secrets From the Dark Side*. Syngress, Rockland, MA, 2004.

[75] J. Stewart. Inside the Storm: Protocols and Encryption of the Storm Botnet. http://www.blackhat.com/presentations/bh-usa-08/Stewart/BH_US_08_Stewart_Protocols_of_the_Storm.pdf, 2008.

[76] Symantec Corporation. The State of Spam A Monthly Report. http://eval.symantec.com/mktginfo/enterprise/other_resources/b-state_of_spam_report_08-2008.en-us.pdf, 2008.

[77] B. Taylor. Sender Reputation in a Large Webmail Service. In *Proc. of 3rd CEAS*, 2006.

[78] The Radicati Group, Inc. Email Statistics Report, 2012-2016. http://www.radicati.com/wp/wp-content/uploads/2012/04/Email-Statistics-Report-2012-2016-Executive-Summary.pdf, 2012.

[79] Ubercart. Ubercart. http://www.ubercart.org.

[80] United States Department of Justice. Two Men Sentenced for Running International Pornographic Spamming Business. http://www.justice.gov/opa/pr/2007/October/07_crm_813.html, 2007.

[81] Visa Inc. Visa Check Card Issuer Authorization Performance Self-Diagnostic Tool. http://www.weknowpayments.com/documents/pdf/Visa_Performance_Tool.pdf, 2008.

[82] Y.-M. Wang, M. Ma, Y. Niu, and H. Chen. Spam Double-Funnel: Connecting Web Spammers with Advertisers. In *Proc. of 16th WWW*, 2007.

[83] D. Watson. All Spammers Go to Hell (posting to funsec list). http://www.mail-archive.com/funsec%40linuxbox.org/msg03346.html, 2007.

[84] T. Wilson. Competition May be Driving Surge in Botnets, Spam. http://www.darkreading.com/document.asp?doc_id=142690, 2008.

[85] M. W. Wong. Sender Authentication: What To Do. http://www.openspf.org/blobs/sender-authentication-whitepaper.pdf, 2004.

[86] X-Cart. X-Cart. http://www.x-cart.com/.

[87] Zen Ventures, LLC. Zen Cart. http://www.zen-cart.com/.

[88] F. Zhou, L. Zhuang, B. Zhao, L. Huang, A. Joseph, and J. Kubiatowicz. Approximate Object Location and Spam Filtering on Peer-to-Peer Systems. In *Proc. of 4th Middleware*, 2003.