UC Riverside

UC Riverside Electronic Theses and Dissertations

Title

Understanding Online Malicious Behavior: Social Malware and Email Spam

Permalink

https://escholarship.org/uc/item/8502w5xx

Author

Huang, Ting-Kai

Publication Date

2013

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA
RIVERSIDE

Understanding Online Malicious Behavior: Social Malware and Email Spam

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Ting-Kai Huang

August 2013

Dissertation Committee:

Dr. Michalis Faloutsos, Co-Chairperson
Dr. Harsha V. Madhyastha, Co-Chairperson
Dr. Srikanth V. Krishnamurthy
Dr. Iulian Neamtiu

Copyright by
Ting-Kai Huang
2013

The Dissertation of Ting-Kai Huang is approved:

Committee Co-Chairperson

Committee Co-Chairperson

University of California, Riverside

Acknowledgments

I would like to express my gratitude to Dr. Michalis Faloutsos and Dr. Harsha V. Madhyatha for their guidance and inspiration throughout my PhD study. Their impact on me is not only on my academic research but also on my attitude towards life. My deepest appreciation goes to my committee, Dr. Srikanth Krishnamurthy and Dr. Iulian Neamtiu, for their valuable advice to help me complete my thesis. I would also like to express my heartfelt appreciation to my parents for supporting my decisions; my sister for sharing her experiences with me whenever I felt perplexed; my brother for taking care of my parents while I pursue my PhD in the United States. Finally I would like to thank my wife, Yi-Ying Lai, without whom none of this would have been possible.

ABSTRACT OF THE DISSERTATION

Understanding Online Malicious Behavior: Social Malware and Email Spam

by

Ting-Kai Huang

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, August 2013
Dr. Michalis Faloutsos, Co-Chairperson
Dr. Harsha V. Madhyastha, Co-Chairperson

Online social networks (OSNs) have become a popular new vector for distribution of malware and spam, which we refer to as *socware* . Unlike email spam – which is mostly sent by spammers directly to intended victims – socware leverages in great part on the "word-of-mouth" propagation of online social networks.

In the dissertation, we study socware through analysis of 173 billions posts collected from 16K users of MyPageKeeper, a Facebook application created by ourselves, for ten months. We uncover not only the ecosystem of socware, but also the incentive mechanisms used by socware to help itself spread through Facebook.

We observe the emergence of AppNets, groups of applications that collaborate in enabling cascades. We found that 44% of cascades are enabled by Facebook applications, i.e., the posts are made by applications that have been installed by luring users. Moreover, we notice that 37% of cascades contains URLs that direct users back to Facebook, either Facebook application installation pages or cooperate pages which want to get an "Like" endorsements.

During the analysis of the provenance of socware posts, we find that more than half of socware cascades were propagate by users clicking Like/Share on websites outside of Facebook, or by users manually posting spam. It not only shows the carelessness of users in online social networks and the importance of user education to eliminate socware from Facebook but also raise our interest about the incentives mechanism used by socware distributors.

Our work provides the first systematic taxonomy of the psychological techniques that socware uses to lure users. We show that socware uses several incentive mechanisms which we group into financial and social incentives. A striking findings is that, much like biological viruses, the most effective socware cascades use multiple incentives to get around our diversity in susceptibility to distinct incentives. Our observations, supported with simulations, show that it an effective strategy to distribute socware in online social networks. Last but not least, we show the very different characteristics of the two generations of online malicious behavior: socware and spam emails. For example, most of spam email focus on earning commission by selling cheap pharmaceutical or counterfeit, socware focus on phishing users' personal information and ask users to complete surveys. Our work is one of the first studies focusing on measuring and modeling socware and the mechanisms, and it can help us better understand its intention and propagation mechanisms that can lead ultimately to stopping it.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since its appearance in 1978, the volume of spam emails has increased rapidly, with recent estimates identifying 107 billion spam email messages globally per day. As a result shown in [9], spam emails cost the society around $20 billion dollars a year, when spammers only earn $200 millions dollars. It makes people eager to stop spammers business. However, the arms race between spammers and anti-spam email e xperts seems to be a never-ending one.

Now, spammers shift their attention from email spam to online social network. This new playground for spammers are filled with people who are not familiar the service and easy to be lured. In 2011, a Facebook malicious application infected 5 million accounts in 48 hours [28]. It becomes a serious problem for online social network service providers.

We use the term socware to describe parasitic or damaging behavior including identity theft, distribution of malicious URLs, spam, and malicious applications that utilizes online social networks. The use of posts from friends adds a powerful element in the propagation of social malware: it comes implicitly with the endorsement of a friend who

allegedly posts the information. As we showed later, socware relies on human incentive mechanisms and cognitive incapability to distinguish between a legitimate request from a friend and a bogus request from an infected friend.

In order to detect socware, we create MyPageKeeper, which is a Facebook security application. After people installed MyPageKeeper, MyPageKeeper monitors users' timeline and newsfeed and warn users by either comment on the posts or email users when it detects any posts with malicious links. The performance and accuracy are evaluated in [103]. With the help of MyPageKeeper, we create a our socware dataset for socware analysis.

In this dissertation, our goal is to understand the new type of online malicious behaviors, socware. Socware is distributed through posts, typically consisting of a short text body and a URL, which appears on users' walls and consequently on news feeds. Grouping post which sharing a single goal into cascades is the first step to understand socware. We use the term cascades instead of campaigns to emphasize the fact that socware cascades through the social network. While the definition of a campaign is relatively well defined in the context of email spam [119, 101, 102, 59, 86, 52], the community's understanding of socware cascades is limited in comparison. With an appropriate grouping mechanism, we further study the spatio-temporal properties of socware and uncover the supporting ecosystem of socware. Next, we analyze the incentive mechanisms used by socware to lure users into their traps. Unlike email spam, which entices users with counterfeits, and adult/sex content, socware targets human's susceptibility of distinct incentives and cognitive capacity to distinguish between a legitimate request from a friend and a bogus request from an infected friend. Finally, we study the differences between socware and email spam.

1.1 Socware Dataset

In this dissertation, we first present MyPageKeeper, a Facebook security application to protect Facebook users from socware. The application launched in July of 2011 and has 16,240 installations until April of 2012. By monitoring these users and their friends, we indirectly monitor 3.9 million users wall. With the help of MyPageKeeper, we build up a valuable Facebook socware dataset that enable our studies.

MyPageKeeper is an accurate, efficient, and scalable. In order to operate MyPageKeeper at scale, but at low cost, the distinguishing characteristic of our approach is our strident focus on efficiency. Prior solutions for detecting spam and malware on OSNs (which we describe in detail later) rely on information obtained either by crawling the URLs included in posts or by performing DNS resolution on these URLs. In contrast, our socware classifier relies solely on the *social context* associated with each post (e.g., the number of walls and news feeds in which posts with the same embedded URL are observed, and the similarity of text descriptions across these posts). Note that this approach means that we do not even resolve shortened URLs (e.g., using services like bit.ly) into the full URLs that they represent. This approach maximizes the rate at which we can classify posts, thus reducing the cost of resources required to support a given population of users.

We employ a Machine Learning classification module using Support Vector Machines on a carefully selected set of such features that are readily available from the observed posts. 97%of posts flagged by our classifier are indeed socware and it incorrectly flags only 0.005% of benign posts. Furthermore, it requires an average of only 46 ms to classify a post.

1.2 Profiling socware cascades

With the help of MyPageKeeper, we are able to understand this new emergent malicious behavior on online social networks. The key question of our work is the need to define, model, and understand the ecosystem that enables cascades on OSNs, specifically on Facebook. Most spam and malware is distributed through posts, which typically consist of a short text body and a URL. These malicious posts, which appear on users' walls and consequently on news feeds, can be grouped into *cascades*; Intuitively, a cascade should capture the posts that are the result of an attempt to achieve a single goal, such as attracting users to a particular website, or selling a particular product. In the context of email spam, the definition of a campaign is relatively well defined [119, 101, 102, 59, 86, 52]. We argue that, in OSNs, defining a cascade is slightly more involved due to the several factors at play: who is launching it, what posts are used, what URL or URLs drive it, and which infrastructure is enabling.

Our key contribution is a systematic study of socware cascades on Facebook by analyzing a five month long trace of roughly 100K spam posts identified from the walls of over 3 million Facebook users. There are three prongs to our approach: (a) defining cascades systematically, (b) studying spatio-temporal properties of cascades, and (c) examining the infrastructure that enables cascades. We discuss our contributions in each of these directions below.

- **A systematic approach to define cascades.** The most direct way to identify cascades is by grouping posts that point to the same *landing URL*. However, the landing web page information may often be unavailable, for reasons that we detail

later. Therefore, we examine how to identify cascades in the absence of landing URLs. We identify six definitions of cascades and show that different definitions can give significantly different results in both the number and the trends of cascades. For example, the number of cascades obtained by grouping together posts with identical spam message content is 13 times higher than the number of cascades obtained by grouping based on `Landing-URL`[1]. Interestingly, hackers often vary the text included in the posts of a cascade, whereas they rarely combine the same text message with different URLs. At the same time, with high statistical accuracy (Adjusted Rand Index > 0.98), we get the same set of cascades by using either URLs included in posts or the landing URLs. Thus, our findings suggest that grouping posts based on the URLs they include is an effective way of identifying cascades (at least until hacker behavior changes).

- **Spatio-temporal characteristics of cascades.** We find that cascades are quite prevalent. Over five months, more than 60% of the monitored users, were exposed to at least one cascade. Interestingly, over the duration of our dataset, we observe an evolving trend in which cascades appear to be less active, probably with the goal of being less conspicuous and thus evading detection. Indeed, hackers appear to be successful in this goal as cascades later in our dataset last longer than those seen in the initial months. Furthermore, we find that users are more likely to receive socware from their most active friends, whereas the number of friends a user has seems to bear little significance to the chances of the user being exposed to socware.

[1]See Section 3.2.2 for more details

- **Forensic analysis of cascades.** We conduct an in-depth forensics analysis of the cascades present in our dataset. We find that over 44% of cascades are launched by Facebook applications, i.e., the posts are made by applications that have been installed by users, who may have been tricked into installing those apps. In addition, we find that over 37% of cascades contain URLs which lead users back to Facebook, either apps to be installed or corporate pages that want to get a "Like" endorsement. These results indicate that a significant fraction of cascades are hosted by Facebook and propagated using Facebook functionality. Further investigation shows that these cascade-enabling Facebook apps form colluding groups. We find at least 144 application groups of size five or larger that form tightly connected collaborative groups in enabling the same cascades.

1.3 Psychology of socware

In recent years, online social network (OSN) users have lived with the threat of malware attacks. Unlike email spam, where the attacker incentivize users to click on an URL with offers such as money or cheap counterfeit products, OSN malware (socware) allows spammers to easily tap into the victims social network.

In this study, we try to uncover incentive mechanisms used by socware to lure users into their traps. We first identify the different categories of incentives. We recruited 305 volunteers from Amazon mechanical turk [2] to classify the incentives of socware cascades. Furthermore, we develop a classifier, which can correctly classifies the incentive of cascades with high F_1 scores [11]. More specifically, we can decide is a post contains social or financial

incentives with 0.83 and 0.88 F_1 scores respectively.

Second, we conduct extensive study with real malicious socware posts, which are collected from roughly 3.5 M users walls over ten months, and arrive at several interesting observations: 1) Most of the cascades we observe contain social incentives. We find that 81.66% of cascades contain social incentives. However, only 29.68% of cascades contain financial incentives. 2) Different incentives show different characteristics. On average, pure social incentives cascades compromised 16.44% more accounts per cascade than financial incentives. However, financial incentives cascades last 58.4% longer than social incentives cascades. 3) Combined/Mixed incentives cascades actually more effective and last longer than pure social/financial incentives cascades.

Finally, We notice that cascades with combined/mixed incentives are the most effective cascades. They, much like biological virus, get around users' diversity in susceptibility to distinct incentives by using multiple incentives into their cascades.. Through simulations we see that even if the susceptibility of combined incentives is less than that of one specialized incentive, combined incentives still can compromised $70\% - 180\%$ more users than specialized incentives.

1.4 Comparisons between socware and email spam

As reported shown in Symantec [50], the rise of socware makes the percentage of spam emails out of all emails drop 13%. There is a very high chance that online social networks become the new main playground of spammers. With our socware dataset in hand, we want to understand the differences between email spam, the major online ma-

licious behavior in the last generation, and socware, the next generation online malicious behaviors. We compare socware with spam emails from three different perspectives: 1) support ecosystem, 2) intention of spamming websites, 3) incentives of spam emails and socware posts.

We summarize the differences from three perspectives here:

1. ***Support Ecosystem*** **The distribution methods are different**. Spammers send spam emails directly to users' inboxes. Socware leverages on the word-of-mouth propagation of online social networks. It has been shown that this propagation method is very infectious. Moreover, we find many socware advertising websites are hosted in Facebook. It makes the traditional blacklist-based protection method ineffective. Compared with email spammers who usually try to avoid exposing their websites in the email directly by setting up redirection hosts in front of their websites, socware developers spend less effort to protect their websites.

2. ***Intention of spamming websites*** Most of websites advertised by spam emails sell authentic goods. The main categories of websites advertised by spam emails are "Pharmaceutical", "Watches/Jewelry" and "Sexual/Dating". However, the websites advertised by socware cascades are "phishing" or "survey scam".

3. ***Incentives of spam emails/socware posts***

   Email spams entice users by cheap goods. However, socware cascades mainly lure users by social incentives. Email spammers usually join affiliate programs and earn money from the commission. As the results, they mainly focus on avoiding spam emails from being detected. As long as spam emails reach users' inboxes, the cheap goods, for

example, drugs, replicas, and computer software, which are sold on the websites will bring users who need them to visit the spam-advertised websites automatically. On the contrary, socware developers exploit users "social curiosity" or "support something you like" to spread their socware.

1.5 Summary

Our research makes three contributions to the study of online malicious behavior: First, we systematically analyze possible grouping methods for socware posts and suggest that grouping socware posts into cascades by URLs embedded in the posts is an appropriate way. We also observe that a big portion of socware posts is actually posted with users awareness, either by users clicking the "Like/Share" on the websites outside of Facebook, or by users posting manually. It shows the importance of user education for socware, like what we do for educating users about spam emails. We further observe the rise of AppNets where malicious Facebook applications collude at large scale. The AppNets might provide spammers a new business model. For example, Facebook page owners might pay for spammers to increase the popularity of their pages. Increase the number of 'Like' of the pages is one way to do that. As reported in [103], several Facebook applications have provided this services to pages.

Second, we study the incentives used by socware developers to promote their socware. We show that most socware posts offer some form of incentive to lure new victims, which we broadly classify as social, financial, or combined (social and financial) incentives. One of our most striking findings is that, much like biological viruses, the most effective

socware cascades use multiple incentives to get around our diversity in susceptibility to distinct incentives.

Finally, we compare socware with spam emails from different perspectives. The difference between socware and spam emails not only exist in the whole spamming infrastructure, no matter in the front **Spam Distribution Infrastructure**, or the back end **Scam Hosting Infrastructure**, but also exist in the intention of spamming websites and the incentives used by spam emails/posts.

Chapter 2

Introduction to MyPageKeeper

In this chapter, we present our experience in developing and deploying MyPageKeeper, a Facebook security app with more than 16K installs and 173 billion posts collected over roughly ten months. After users install it, MyPageKeeper monitors their walls and news feeds, detects and flags malicious posts. MyPageKeeper was officially launched in June 2011 and attracted significant attention from the electronic and printed press.

2.1 Introduction of Socware

Before we describe what is the design of MyPageKeeper and how we implement it, we will start by the definition of *socware*. In the later of the dissertation we will show that *socware* has significantly different characteristics compared to traditional email spam, phishing, and web malware.

What is socware?

We define socware as fake, annoying, possible damaging posts from friends of

the potential victim. In the context of this dissertation, we consider a Facebook post as malicious, if it satisfies one of the following conditions:

1. The post spreads malware and compromises the device of the user.
2. The advertised web page requires the user to give away personal information.
3. The post promises false rewards (e.g., free products).
4. The post is made on a user's behalf without the user's knowledge
5. The web page pointed to by the post requires the user to carry out tasks (e.g, fill out surveys) that help profit the owner of that website, or
6. The post causes the user to artificially inflate the reputation of the page (e.g., by forcing the user to 'Like' the page).

While the first two criteria are typical malware and phishing, the latter four are distinctive of socware. *Note that,* as with email spam, there can be some ambiguity in the definition of socware: a post considered as annoying by one user may be considered useful by another user. In practice, our ultimate litmus test is the opinion of MyPageKeeper's users: if most of them report a post as annoying, we will flag it as such.

How does socware work?

Socware appears in a Facebook user's timeline or news feed typically in the form of a post which contains two parts. First, the post contains a URL to a webpage that hosts either malicious or spam content. Second, socware posts usually contain a catchy text message (e.g. *"two free Southwest tickets"*) that entice users to click on the URL included in the post. Optionally, socware posts also contain a thumbnail screenshot of the landing

page of the URL, also used to entice the user to click on the link. For example, a purported image of Osama's corpse is included in a post that claims to point to a video of his death. This part of information is extracted from the advertised website directly by Facebook. From our dataset, we also identify a non-trivial part of socware posts which rely on these information to lure users into their traps.

Once a user follows the embedded URL to the target website, the post tries to propagate itself through that user. For this, the user is often asked to complete several steps in order to obtain the fake reward (e.g., "Free Facebook T-shirt"). These steps involve "Liking" or "sharing" the post, or posting the socware on the user's wall. Thus, the cycle continues with the friends of that user, who see the post in their news feed. In contrast, users seldom forward email spam to their friends.

The exploitation often starts after the propagation phase. The hacker attempts either to learn the user's private information via a phishing attack [12], to spread malware to user devices, or to make money by "forcing" a particular user action or response, such as completing a survey for which the hacker gets paid [25].

Where is socware hosted?

Socware can be classified into two categories based on the hosting infrastructure.

- **Socware hosted outside Facebook:** in this category, URLs point to a domain hosted outside Facebook. Since the URL points to a landing page outside the OSN, hackers can directly launch the different kinds of attacks mentioned above once a user visits the URL in a socware post. Though several URL blacklists should be able to flag such URLs, the process of updating these blacklists is too slow to keep up with the viral propagation of socware on OSNs [76].

- **Socware hosted on Facebook:** a significant fraction of socware is hosted on Facebook itself: the embedded URL points to a Facebook page or application. Naturally, current blacklists and reputation-based schemes fail to flag such URLs. The different type of Facebook objects are: **1)Malicious Facebook applications**. Malicious applications post catchy messages (e.g., *"Check who deleted you from your profile"*) on the timelines of users with a link pointing to the installation page of the application. Users are conned into installing the application to their profile and granting several permissions to it. The application then not only gets access to that user's personal information (such as email address, home town, and high school) but also gains the ability to post on the victim's timeline. As before, posts on a user's timeline typically appear on the news feeds of the user's friends, and the propagation cycle repeats. Creating such applications has become easy with ready to use toolkits starting at $25 [23]. **2) Malicious Facebook events.** Hackers create Facebook events that contain a malicious link. One such event is the *"Get a free Facebook T-Shirt (Sponsored by Reebok)"* scam. This event page states that 500,000 users will get a free T-shirt from Facebook. To be one among those 500,000 users, a user must attend the event, invite her friends to join, and enter her shipping address. **3) Malicious Facebook pages.** Hackers create a Facebook page and post socware links on the page [34]. We also identified a trend in aggressive marketing by companies that force users to click "Like" on their Facebook pages to spread their pages as well as increase the reputation of the page.

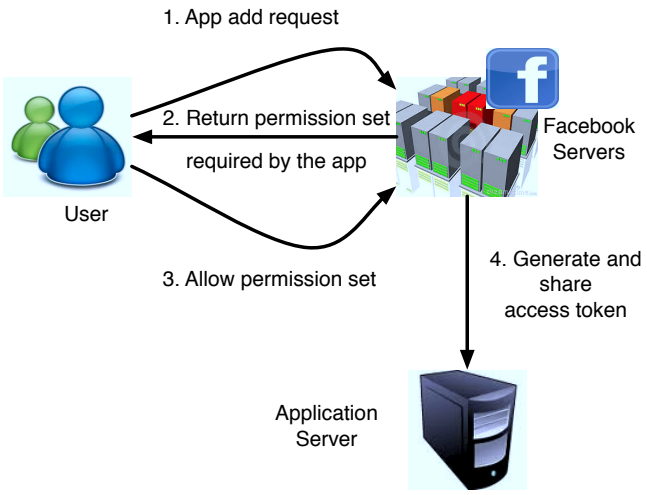


Figure 2.1: Application installation process on Facebook.

2.2 Architecture of MyPageKeeper

When we design MyPageKeeper, we want to make it *accurate*, *scalable* and *efficient*. We are faced with the obvious trade-off between missing malware posts (false negatives), and "crying wolf" too often (false positives). Although one could argue that minimizing false negatives is more important, users would abandon overly sensitive detection methods, as recognized by the developers of Facebook's Immune System [107].

We also want to provide protection from socware for all users on Facebook, not just for a select few. Therefore, the system must be scalable to easily handle increased load imposed by a growth in the number of subscribed users. Finally, we seek to minimize our costs in operating MyPageKeeper. The period between when a post first becomes visible to a user and the time it is checked by MyPageKeeper represents the window of vulnerability

when the user is exposed to potential socware. To minimize the resources necessary to keep this window of vulnerability short, MyPageKeeper's techniques for classification of posts must be efficient.

2.2.1 MyPageKeeper Components

We introduce the six main functional modules of MyPageKeeper here.

- **User authorization module.** We obtain a user's authorization to check her wall and news feed through a Facebook application, which we have developed. Once a user installs the MyPageKeeper application, we obtain the necessary credentials to access the posts of that user. For alerting the user, we also request permission to access the user's email address and to post on the user's wall and news feed. Figure 2.1 shows how a Facebook user authorizes an application.

- **Crawling module.** MyPageKeeper periodically collects the posts in every user's wall and news feed. As mentioned previously, we currently focus only on posts that contain a URL. Apart from the URL, each post comprises several other pieces of information, such as a text message associated with the post, the user who made the post, number of comments and Likes on the post, and the time when the post was created.

- **Feature extraction module.** To classify a post, MyPageKeeper evaluates every embedded URL in the post. Our key novelty lies in considering only the social context (e.g., the text message in the post, and the number of Likes on it) for the classification of the URL and the related post. Furthermore, we use the fact that we are observing more than one user, which can help us detect an epidemic spread.

- **Classification module.** The classification module uses a Machine Learning classifier based on Support Vector Machines, but also utilizes several local and external whitelists and blacklists that help speed up the process and increase the overall accuracy. The classification module receives a URL and the related social context features extracted in the previous step. Since the classification is our key contribution, we discuss this in more detail in Section 2.2.2. If a URL is classified as socware, all posts containing the URL are labeled as such.

- **Notification module.** The notification module notifies all users who have socware posts in their wall or news feed. The user can currently specify the notification mechanism, which can be a combination of emailing the user or posting a comment on the suspect posts. In the future, we will consider allowing our system to remove the malicious post automatically, but this can create liabilities in the case of false positives.

- **User feedback module.** Finally, to improve MyPageKeeper's ability to detect socware, we leverage our user community. We allow users to report suspicious posts through a convenient user-interface. In such a report, the user can optionally describe the reason why she considers the post as socware.

2.2.2 Identification of Socware

The key novelty of MyPageKeeper lies in the classification module (summarized in Figure 2.2). The input to the classification module is a URL and the related social

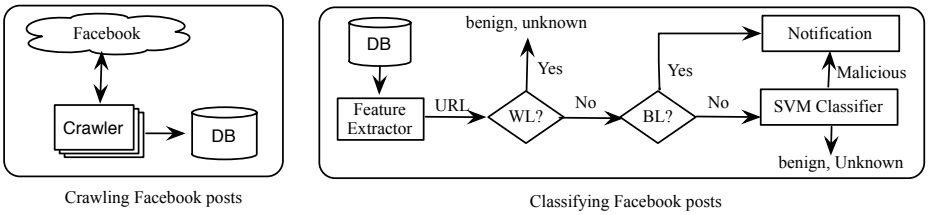


Figure 2.2: Architecture of MyPageKeeper.

context features extracted from the posts that contain the URL. Our classification algorithm operates in two phases, with the expectation that URLs and related posts that make it through either phase without a match are likely benign and are treated as such.

Using whitelists and blacklists. To improve the efficiency and accuracy of our classifier, we use lists of URLs and domains in the following two steps. First, MyPageKeeper matches every URL against a whitelist of popular reputable domains. We currently use a whitelist comprising the top 70 domains listed by Quantcast, excluding domains that host user-contributed content (e.g., OSNs and blogging sites). Any URL that matches this whitelist is deemed safe, and it is not processed further.

Second, all the URLs that remain are then matched with several URL blacklists that list domains and URLs that have been identified as responsible for spam, phishing, or malware. Again, the need to minimize classification latency forces us to only use blacklists that we can download and match against locally. Such blacklists include those from Google's Safe Browsing API [22], Malware Patrol [29], PhishTank [33], APWG [4], SpamCop [37], joewein [26], and Escrow Fraud [10]. Querying blacklists that are hosted externally, such as SURBL [39], URIBL [42] and WOT [44], will introduce significant latency and increase MyPageKeeper's latency in detecting socware, thus inflating the window of vulnerability.

Any URL that matches any of the blacklists that we use is classified as socware.

Using machine learning with social context features. All URLs that do not match the whitelist or any of the blacklists are evaluated using a Support Vector Machines (SVM) based classifier. SVM is widely and successfully used for binary classification in security and other disciplines [97, 89] [40]. We train our system with a batch of manually labeled data, that we gathered over several months prior to the launch of MyPageKeeper. For every input URL and post, the classifier outputs a binary decision to indicate whether it is malicious or not. Our SVM classifier uses the following features: *Spam keyword score, Message similarity, News feed post and wall post count, Like and comment count* and *URL obfuscation,* More details about each feature can be found in [103].

2.2.3 Implementing MyPageKeeper

We provide some details on MyPageKeeper's implementation.

Facebook application. First, we implement the MyPageKeeper Facebook application using FBML [18]. We implement our application server using Apache (web server), Django (web framework), and Postgres (database). Once a user installs the MyPageKeeper app in her profile, Facebook generates a secret access token and forwards the token to our application server, which we then save in a database. This token is used by the crawler to crawl the walls and news feeds of subscribed users using the Facebook open-graph API. If any user deactivates MyPageKeeper from their profile, Facebook disables this token and notifies our application server, whereupon we stop crawling that user's wall and news feed.

Crawler instances and frequency. We run a set of crawlers in Amazon EC2 instances to periodically crawl the timelines and news feeds of MyPageKeeper's users. The

set of users are partitioned across the crawlers. In our current instantiation, we run one crawler process for every 1,000 users. Thus, as more users subscribe to MyPageKeeper, we can easily scale the task of crawling their timelines and news feeds by instantiating more EC2 instances for the task. Our Python-based crawlers use the open-graph API, incorporating users' secret access tokens, to crawl posts from Facebook. Once the data is received in JSON format, the crawlers parse the data and save it in a local Postgres database.

Currently, as a tradeoff between timeliness of detection and resource costs on EC2, we instantiate MyPageKeeper to crawl every user's timeline and news feed once every two hours. Every couple of hours, all of our crawlers start up and each crawler fetches new posts that were previously not seen for the users assigned to it. Once all crawlers complete execution, the data from their local databases is migrated to a central database.

Checker instances. Checker modules are used to classify every post as socware or benign. Every two hours, the central scheduler forks an appropriate number of checker modules determined by the number of new URLs crawled since the last round of checking. Thus, the identification of socware is also scalable since each checker module runs on a subset of the pool of URLs. Each checker evaluates the URLs it receives as input—using a combination of whitelists, blacklists, and a classifier—and saves the results in a database. We use the libsvm [61] library for SVM based classification. Once all checker modules complete execution, notifiers are invoked to notify all users who have posts either on their timelines or in their news feeds that contain URLs that have been flagged as socware.

2.3 Summary

In this chapter, we describe the architecture of MyPageKeeper. At first, we introduce socware, a new type of malware in online social networks, including what it is, how it works and where it is hosted. Then we describe the six basic modules of MyPageKeeper and some implementation details. We don't show the evaluation results of MyPageKeeper in detail, which could be found in [103], because they are not the focus of this dissertation. Instead, we summarize the performance of MyPageKeeper here. 97% of posts flagged by MyPageKeeper are indeed socware and it incorrectly flags only 0.0005% of benign posts. Further, it requires an average of only 46 ms to classify a post.

The dataset used and analyzed in this dissertation are collected by MyPageKeeper. Understanding how MyPageKeeper works can help readers have a better understand about the data we analyze.

Chapter 3

Profiling socware cascades

Motivated by the significant amount of time that users spend on online social networks (OSNs) (e.g., users in the US spend over one-fourth of their time on the Internet on OSNs [3]), spammers have shifted their attention from email to OSNs [43, 82, 47]. Unlike email spam, which is mostly sent by spammers directly to intended victims [1], *socware* [2] leverages the viral propagation on OSNs; every user who falls victim to a socware campaign unknowingly exposes her friends to the same. Thus, socware *cascades* through the social network, and since users receive it with the implicit recommendation of a friend, it is more powerful than email spam.

The goal of our work is to understand the ecosystem that enables socware cascades, specifically on Facebook. Socware is distributed through posts, typically consisting of a short text body and a URL, which appear on users' walls and consequently on news feeds (as we explain in more detail later). As the posts that attempt to achieve a single goal

[1] In 2011, 81.2% of spam emails were sent by botnets [49].

[2] We use the new term socware to refer to social malware [24].

(e.g., redirect users to the same malicious website), cascade through the OSN, hackers can modify the text and/or the URL included in these posts. While the definition of a campaign is relatively well defined in the context of email spam [119, 101, 102, 59, 86, 52], the community's understanding of socware cascades is limited in comparison.

So far, there have been few studies that focus on socware, most of them focus on Twitter, and there has been limited investigation of the underlying enabling ecosystem. Most prior work focuses on detection: identifying spam and malicious posts, and pinpointing accounts used by spammers in Twitter [54, 121]. Whereas recent studies [72, 103] have shown that a large fraction of OSN spam is in fact propagated by legitimate users whose accounts are compromised, e.g., because they installed a malicious Facebook app.

Our key contribution is a systematic study of socware cascades on Facebook [3] by analyzing a five month long trace of roughly 100K spam posts identified from the walls of over 3 million Facebook users through 15K users that have installed our application (the monitored *newsfeed* of a profile sees the posts on the wall of all its friends). Our work consists of three thrusts: (a) studying spatio-temporal properties of socware cascades, (b) examining the infrastructure and mechanisms that enable socware, and (c) analyzing the social engineering tricks that socware leverages. We discuss our contributions in each of these directions below.

a. Spatio-temporal characteristics. We find that socware cascades are quite prevalent. Over five months, more than 60% of the monitored users were exposed to at least one cascade. Interestingly, over the duration of our dataset, we observe an evolving trend in which the median cascade appears to be less active, probably with the goal of being

[3] We do not claim any contribution towards detecting socware, which we consider part of the input for this paper.

less conspicuous and thus evading detection. Indeed, hackers appears to be successful in this goal as cascades later in our dataset last longer than those seen in the initial months. Furthermore, we find that users are more likely to receive socware from their most active friends, whereas the number of friends of a user seems to bear little significance to the chances of the user being exposed to socware.

b. Forensic analysis of cascades. We observe the emergence of AppNets, groups of apps that collaborate in enabling cascades. AppNets seem to play the role that botnets play in the email spam campaigns. Specifically, we find that over 44% of cascades are enabled by Facebook applications, i.e., the posts are made by applications that have been installed by users, who may have been tricked into installing those apps. In quantifying the parasitic symbiosis, we find that over 37% of cascades contain URLs which lead users back to Facebook, either to app installation pages or to corporate pages that want to get a "Like" endorsement. Interestingly, these socware-enabling Facebook apps form colluding groups. Collaboration is prevalent as we find at least 144 tightly-connected application groups of five or more apps enabling the same cascades.

c. Social engineering tricks and intents. We investigate how socware entices and fools users, and we identify two dominant methods: (a) exploiting users' social curiosity (e.g., claiming to show which friends deleted a user or viewed the user's profile), and (b) offering fake free or cool products (e.g., free iPads or free Facebook T-shirts). In our dataset, these two methods account for 46% and 32% respectively of socware cascades seen across at least 10 users. In addition, we observe that over 85% of socware cascades are intended to enable phishing, survey scams, or fraudulent inflation of web page reputation. By contrast, email spam uses significantly different "hooks" (e.g., appeals to sexual curiosity) and has

different intents (e.g., sell low cost pharmaceuticals) [49].

Implications for anti-socware efforts. Our findings could provide useful guidelines for reducing socware on Facebook.

First, our results highlight the importance of focusing specifically on the identification of malicious Facebook applications, particularly groups of coordinating applications that enable socware cascades. Moreover, Facebook can greatly reduce socware on its platform simply by better policing of the content that it admits to be hosted under its domain, in the form of pages, events, etc.

Second, our results show that key enablers for socware are users falling prey to scams that offer fake rewards (such as free products) or that appeal to the social curiosity of users. Thus, better education of users remains a significant stumbling block towards eliminating the threat of socware.

3.1 Background

We present an overview of Facebook terminology and describe the socware dataset that we use.

3.1.1 Facebook

Facebook is the most popular online social network today, with over a billion users, more than half of whom log in to the site daily [13]. On Facebook, information is shared in the form of *posts*. There are four main types of posts on Facebook—*STATUS*, *LINK*, *PHOTO*, and *VIDEO* [7]. In our analysis, we use the following set of features associated with any post:

- *ID*: a unique Facebook-assigned identifier for the post
- *message*: the text message included in the post
- `Posted-URL` or `P-URL`: A URL included in the post
- *create time*: time at which the post was created
- *application*: application that posted on behalf of the user

In addition, a few other fields are included for certain types of posts, e.g., *LINK* posts include a preview of the web page being linked to. Figure 3.1 presents a breakdown of some of the features associated with an example Facebook post.

When a user clicks on the link included in a post, we refer to the URL of the web page that the user will be led to as the `Landing-URL` of the post. The `Landing-URL` for a post may not always be identical to the `Posted-URL`, e.g., the `Posted-URL` may either be a shortened URL (using a URL shortening service such as *bit.ly*) or there may be a chain of HTTP-based or Javascript-based redirections from the `Posted-URL` to the `Landing-URL`. When a `Posted-URL` is shortened using a URL shortening service, we resolve the first redirect of the `Posted-URL` and refer to the URL obtained as the `Resolved-URL`. We refer to the top two-level domain of the `Resolved-URL` as `Resolved-Domain`.

3.1.2 Dataset

Detecting socware. We study the characteristics of socware cascades on Facebook by examining a dataset of malicious posts identified by MyPageKeeper [103], our Facebook security application. MyPageKeeper, which has over 15K subscribed users, continually scans the wall and news feed of every subscribed user to detect malicious posts.

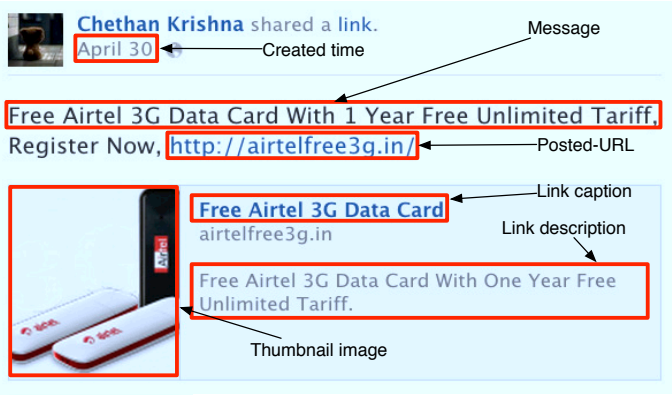


Figure 3.1: Example Facebook post, with its components highlighted.

Since a Facebook user's news feed contains posts made by her friends, the 15K users subscribed to MyPageKeeper enable it to monitor posts seen on the walls of over 3 million Facebook users. The news feed of every MyPageKeeper user contains a subset of the posts made on the walls of the user's friends; the subset of posts from a friend's wall that appear on a user's news feed are selected by Facebook[46].

MyPageKeeper evaluates every URL that it sees on any user's wall or news feed to determine if that URL points to socware. MyPageKeeper classifies a URL as socware if it points to a web page that 1) spreads malware, 2) attempts to "phish" for personal information, 3) requests the user to carry out tasks (e.g., fill out surveys) that profit the owner of the website, 4) promises false rewards, or 5) attempts to entice the user to artificially inflate the reputation of the page (e.g., forcing the user to 'Like' the page to access a false reward). MyPageKeeper evaluates each URL using a machine learning based classifier which leverages the *social context* associated with the URL. For any particular URL, the features used by the classifier are obtained by combining information from all posts (seen across users) containing that URL. Example features used by MyPageKeeper's classifier

Dataset	# of posts with URLs	# of malicious posts	# of unique socware URLs
D-Aug	12,436,634	9,718	452
D-Sep	9,295,575	12,777	613
D-Oct	10,103,378	9,070	275
D-Nov	7,974,616	9,851	234
D-Dec	11,373,501	16,045	347
D-Tot	71,861,393	99,935	2,419
D-Land	—	72,702	1,338

Table 3.1: Summary of datasets.

include the similarity of text message across posts and the number of comments/Likes on those posts. MyPageKeeper has false positive and false negative rates of 0.005% and 3%. For more details about MyPageKeeper's implementation and accuracy, we refer interested readers to [103]. In this work, we analyze socware identified by MyPageKeeper with the goal of analyzing the characteristics of how socware cascades through Facebook, so as to help improve anti-socware measures such as MyPageKeeper.

Datasets. In this chapter, we analyze a dataset of 117 million posts [4] examined by MyPageKeeper over the course of five months (August to December 2011), of which it flagged 99,935 posts as socware. For each post in this dataset, apart from the above mentioned set of features, the dataset also specifies the set of users on whose news feeds this post appeared. We refer to this complete dataset as the *D-Tot* dataset.

Next, in order to analyze malicious activities on a monthly basis—both to identify trends that hold in every month and to identify the evolution of hackers—we partition the

[4]MyPageKeeper only examines posts that contain URLs (including obfuscated URLs), since posts that do not contain any links typically do not correspond to socware.

D-Tot dataset into one dataset for each month. Since the number of users subscribed to MyPageKeeper varies across months, we partition the dataset in such a manner so as to make the datasets across months comparable. To this end, we randomly sample the same number (5K) of users for every month. We then create a dataset for each month as the set of posts collected from the walls and news feeds of the 5K users selected for that month. We thus create five datasets—*D-Aug*, *D-Sep*, *D-Oct*, *D-Nov*, *D-Dec*—corresponding to the five months in our dataset.

During MyPageKeeper's operation, every 12 hours, it crawls the URLs in all posts flagged as socware in the last 12 hours. We limit our crawling frequency due to resource limitations. For every URL crawled, it records the landing URL, its IP address, other *WHOIS* information of the landing domain, and contents of the landing page. It also records the redirection chain of URLs and IPs linking the URL crawled to the landing URL. However, even though MyPageKeeper crawled every URL within 12 hours of it being spotted on Facebook, only $1,338$ of the $2,419$ unique URLs seen in the *D-Tot* dataset were still active when they were crawled. Our *D-Land* dataset comprises that subset of malicious posts from the *D-Tot* dataset which contain URLs that MyPageKeeper was successfully able to crawl. Table 3.1 summarizes all of our datasets.

Representativeness. MyPageKeeper monitors posts on the walls of roughly 3M Facebook users —a small fraction of Facebook's billion users. Therefore, it is hard to gauge the representativeness of our results for all of Facebook. Furthermore, the fact that security conscious users are more likely to install MyPageKeeper could introduce potential bias. However, we find that the subset of the social graph comprising MyPageKeeper users and their friends contains 977 connected components, which implies that they do not seem to

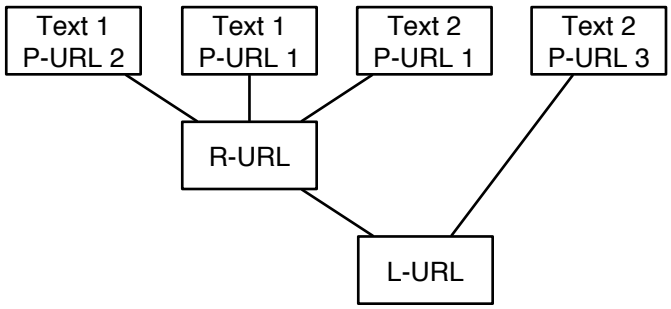


Figure 3.2: An example cascade that seeks to lure users to the web page hosted at *L-URL*. The text message and URLs included in posts, and the URLs that they resolve to, can vary across posts in a cascade.

belong to a closely knit community. Furthermore, since all of our analyses are based on socware observed on the walls of a sufficiently large number of users, we believe that our results have significant takeaways for anti-socware measures (see Section 3.5). Moreover, where appropriate, we demonstrate the statistical significance of our results by analyzing a subset of our data and showing that the result extrapolates to all of our users.

3.2 Cascade Definitions

Our analysis of our socware dataset begins with the identification of spam cascades. We describe the challenges in doing so and the procedure we follow to address them here.

3.2.1 Overview of cascades

We first consider how a hacker may conduct a cascade on Facebook. First, the hacker will have to create a web page to which he wishes to attract users. This web page could take advantage of visiting users in several ways, such as asking users to divulge

private information (e.g., credit card number) or requesting users to fill out a survey (for the monetary benefit of the hacker) in exchange for "free" products. To lure users to these web pages, the hacker may propagate various types of posts on Facebook or any other OSN.

As shown in Figure 3.2, which shows a cascade with the landing web page *L-URL*, the posts in a cascade may vary either in the text message or the URL included in the post. In our example, some of the posts leading to the web page *L-URL* include the text message *Text1* while others include the text message *Text2*. The hacker may include different text messages in different posts either in order to appeal to varying user types or to evade easy detection (see Table 3.3 for examples). On the other hand, the hacker can also vary the `Posted-URL` across posts; this ensures that at least some of the users exposed to these posts can reach the target web page even if a subset of the URLs included in the posts is blacklisted. In our example, even if the `Posted-URL` *P-URL1* is blacklisted, users exposed to posts containing *P-URL2* can still reach the target web page. To further make the cascade resilient to URL blacklisting, the hacker can use a chain of redirection URLs between the URLs included in the post and the target web page, e.g., *P-URL1* and *P-URL2* resolve to *R-URL1*, which in turn redirects to *L-URL*.

In our study, we consider every distinct `Landing-URL` to correspond to a unique cascade. While it is possible that a hacker may create multiple web pages for a single cascade (e.g., to be able to tolerate blacklisting of a subset of these web pages), detecting such cases in an automated manner is tricky. For example, in our dataset, we find 37 landing pages that all have over 95% color histogram similarity with each other. Yet, manual inspection reveals that all of these correspond to Facebook application installation pages, but for completely unrelated applications. Further, a hacker could place the same content on multiple web

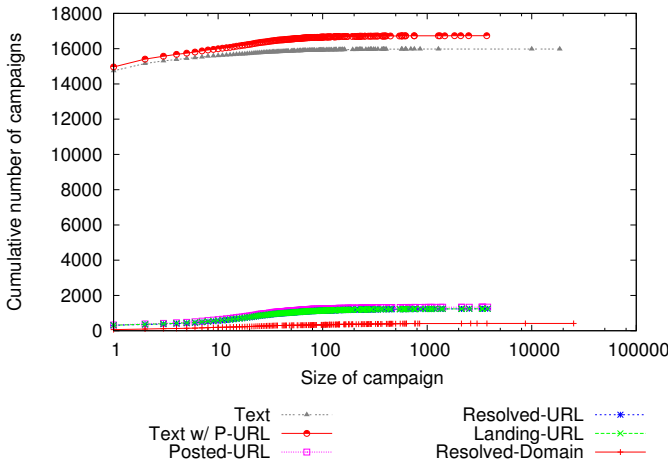


Figure 3.3: Distribution of cascade size based on different definitions of cascades.

pages with different stylistic themes (e.g., host the same survey on two web pages with different background images). We defer the identification of similar landing pages to future work.

3.2.2 Cascade definitions

Although one can identify cascades by grouping together posts that lead to the same target web page, `Landing-URL` information may not always be available for several reasons. On the one hand, web pages hosted by hackers may get blacklisted and therefore they may be no longer available at the time of analysis. For example, as mentioned earlier in the description of our dataset, web pages for less than 55.31% of URLs included in malicious posts existed 12 hours after they appeared on Facebook. On the other hand, it may not be an effective strategy to rely on access to target web pages in order to identify cascades, because hackers could detect that their web page is being accessed for analysis purposes (and not by a normal Facebook user), and therefore return different web page

content selectively. Furthermore, crawling web pages en masse is not efficient.

For these reasons, we consider the following question here: *how do we group spam posts into cascades simply based on the basic features associated with every post?* To answer this question, we use the features associated with Facebook posts to define the following six ways of grouping posts into cascades:

- `Text w/ P-URL`: All posts within a cascade are exactly identical, i.e., they have the same text message and posted URLs.
- `Text`: Posts must share exactly the same text message, but may differ in the posted URLs.
- `Posted-URL`: All posts in a cascade share the same posted URLs.
- `Resolved-URL`: Posts in the same cascade share the same resolved URLs.
- `Resolved-Domain`: Two posts are in the same cascade if their resolved URLs are in the same top two-level domain.
- `Landing-URL`: All posts in a cascade have the same landing URLs.

3.2.3 Impact of cascade definition

First, we examine the effect that different cascade definitions have on the distribution of two common metrics used to study socware cascades—cascade sizes (i.e., number of posts in a cascade) and the activity of cascades (i.e., the number of active cascades as a function of time). To make different cascade definitions comparable, we use the *D-Land* dataset in this part of our analysis.

Table 3.2 shows the number of cascades seen when using the six cascade definitions

Cascade definition	# cascades	Mean cascade size	Variance of cascade size
`Text w/ P-URL`	16,771	4.33	2,420.77
`Text`	15,786	4.61	35,033.78
`Posted-URL`	1,338	54.34	59,003.21
`Resolved-URL`	1,246	58.35	63,195.85
`Landing-URL`	1,247	58.30	63,384.63
`Resolved-Domain`	415	175.19	1,782,650.18

Table 3.2: Distribution of cascade sizes in the *D-Land* dataset when using different ways of grouping posts into cascades.

considered. Note that the total number of posts is the same in all cases. We can see that the manner in which posts are grouped into cascades has a significant impact on the number of cascades. For example, we see $16,771$ unique cascades when using `Text w/ P-URL`-based cascades, which is over $35x$ times the number of cascades observed with the `Resolved-Domain` definition.

Further, we inspect the distribution of cascade sizes when using different cascade definitions. A cascade's size is a good measure of its reach or effectiveness. A cascade that spread virally will have a much larger number of posts in our dataset compared to a cascade that was not as effective. Note that, in the case of socware, hackers rely on victim users to help spread the cascade, e.g., users are lured into "sharing" the malicious post or installing an application controlled by the hacker. In contrast, in the case of email spam, hackers can directly send spam email to all targeted users.

Figure 3.3 shows the distribution of cascade sizes for different cascade definitions,

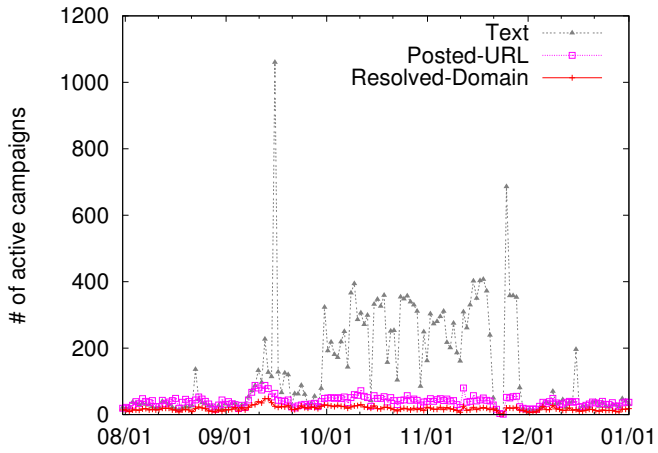


Figure 3.4: Number of active cascades over time by using different cascade definitions. For clarity, we omit `Text w/ P-URL`, whose curve is similar to that for `Text`, and `Resolved-URL` and `Landing-URL`, whose curves are similar to that for `Posted-URL`.

while Table 3.2 presents the mean and variance of cascade size distribution based on each definition. The figure shows that we can roughly divide our six definitions of cascades into three groups based on the similarity in cascade size distributions: 1) `Text` and `Text w/ P-URL`, 2) `Posted-URL`, `Resolved-URL`, and `Landing-URL`, 3) `Resolved-Domain`. The average cascade size in the first group is roughly 11 times larger than that in the second group.

Next, we study the impact of different cascade definitions on the number of active cascade over time, which represents the intensity of malicious activities. From Figure 3.4, we see that different definitions result in significantly different distributions for the number of active cascades over time. When identifying cascades based on the similarity of text messages across posts, we find that the number of active cascades fluctuates heavily over time. However, the curves of URL-based cascade definitions are relatively stable.

3.2.4 Variation in text messages

Based on the above results, we see that, with cascades identified based on the similarity of the text included in posts (`Text` cascades), 1) there are several spikes in the intensity of socware over time, as seen by the significant variance in the number of active cascades in Figure 3.4, and 2) a large fraction of cascades include a single post, as seen in Figure 3.3. Upon manual inspection of malicious posts, we find that this is because hackers often vary the text message included in different posts that advertise the same URL. They seem to do so either 1) to appeal to users by including users' names in the posts, or 2) to obfuscate the text and evade detection.

Table 3.3 shows text messages from three example cascades in which several posts advertised the same URL with variations in the included text. In *Cascade 1*, the username of the recipient of the malicious post is embedded in the post to make the message more convincing; the `Posted-URL` was the same in all posts that included variations of this text message. In contrast, *Cascade 2* shows hackers varying messages in such a manner that they appear visually similar to users and yet evade automated detection. While similarity of text across posts can potentially be detected in these two cases, that is not always the case. *Cascade 3* shows an example where the text message across different posts in a cascade bear little similarity. While these results show that hackers appear to often vary the text message included in various posts that include the same URL, we next investigate how this variation in text message compares to the variation in `Posted-URL` across posts that include the same text message. For every `Text` (`Posted-URL`) cascade, we count the number of distinct posted URLs (text messages) observed across posts in that cascade.

Cascade 1	***angela*** *guess what? i just got a new dell xps 1530 laptop to test out and keep without paying anything! i doubt this will last long, dell is only sending out a limited supply of promo laptops in each area so i would hurry. let me know when you get one to.*
Cascade 2	***WOW l can't*** *believe that* ***u*** *can check who* ***ls viewing*** *ur profile! l just checked my TOP* ***profile*** *visitors and l am trully SHOCKED at who* ***is*** *seeing my* ***profile!*** *You can also see* ***WHO*** *VIEWED YOUR PROFlLE here.*
	W0W I cant *believe that* ***you*** *can check who* ***is vievving*** *ur profile! I just checked my TOP* ***proffile*** *lookers and I am trully SHOCKED at who* ***ls*** *viewing my* ***proffile*** *You can also see* ***VVHO*** *VIEWED YOUR PROFlLE here.*
Cascade 3	*wow, got a super free sexy t-shirt from facebook team...grab ur's from here...fb team is giving free t-shirts :d :-*
	*facebook gifting free tee's to all users, just verify ur account & grab the super cool t-shirt! link : d <3 ps : got mine ^_^*

Table 3.3: Example cascades in which hackers vary text messages across posts.

Figure 3.5 plots this distribution across `Text` and `Posted-URL` cascades. We find that a `Text` cascade has only 1.06 posted URLs on average, with only 1.39% of `Text` cascades advertising multiple posted URLs. In contrast, we find that every `Posted-URL` cascade includes 12.53 distinct text messages on average, with over 28.15% of `Posted-URL` cascades including posts with varying text messages. This shows that hackers often vary the text message used to advertise a URL, but rarely use the same text message to advertise multiple URLs.

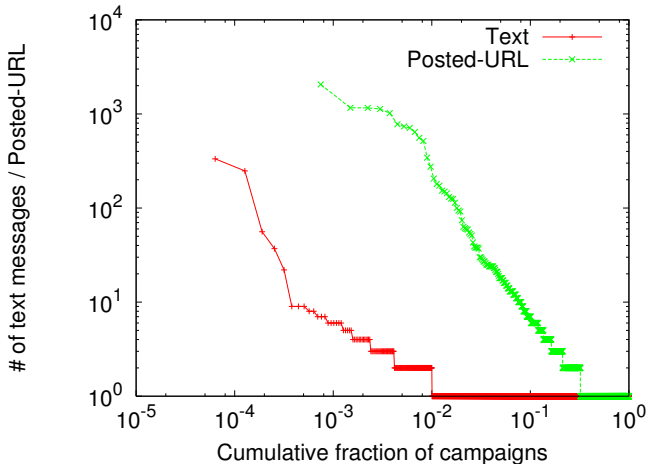


Figure 3.5: The contrast between the use of text messages and posted URLs in socware cascades.

3.2.5 Equivalence of definitions

Our results thus far indicate that the manner in which posts are grouped into cascades strongly impacts the inferred distribution of cascade sizes and number of active cascades. Therefore, one cannot simply identify cascades in an ad-hoc manner. However, we also note that some subsets of the six cascade definitions that we consider result in similar distributions with respect to cascade size and activity. For example, we find the distributions to be similar when grouping posts into cascades either based on the posted or landing URLs.

We next attempt to confirm the similarity, or equivalence, of different cascade definitions. To do so, we consider every pair of cascade definitions and compare the set of cascades obtained with the two definitions. Note that the total number of posts is the same irrespective of how cascades are defined. Hence, the set of cascades obtained with any particular cascade definition can be thought of as a different clustering of the total set of

Definition	Text	Text w/ P-URL	Posted-URL	Resolved-URL	Landing-URL	Resolved-Domain
Text	-	[0.16,0.79]	[0.16, 0.28]	[0.16, 0.23]	[0.01, 0.29]	[0.20,0.53]
Text w/ P-URL		-	[0.26, 0.99]	[0.26, 0.99]	[0.26, 0.96]	[0.01, 0.35]
Posted-URL			-	**[0.96, 1.0]**	**[0.98, 1.0]**	[0.07, 0.34]
Resolved-URL				-	**[0.95, 1.0]**	[0.07, 0.36]
Landing-URL					-	[0.04, 0.49]
Resolved-Domain						-

Table 3.4: ARI values between different cascade definitions. Minimum and maximum values across months are shown. We compare Landing-URL with other definitions based on posts which contain landing URLs in monthly datasets.

posts.

To compare the clustering produced by a pair of cascade definitions, we use the *Adjusted Rand Index (ARI)* metric [100]. The ARI metric compares two different ways of clustering the same data by counting the number of pairs that are in the same cluster in one case but not in the other, or vice-versa, and computing the fraction that these represent out of the total number of pairs. ARI values vary from 0 to 1 and the greater its value the greater the similarity of clusterings being compared. In our use case, the ARI metric compares two cascade definitions by considering all pairs of posts and counting the fraction that are in the same cascade with one definition but in different cascades with the other.

The ARI values for all pairs of cascade definitions are listed in Table 3.4. For each pair of cascade definitions, we list the minimum and maximum value of ARI seen across the five months in our data. First, we find that Posted-URL cascades are highly similar to those based on Resolved-URL and Landing-URL; the ARI value is greater than 0.96 in

all months. Note that this high degree of similarity is seen over a dataset which includes over 58 posts on average for every `Landing-URL` cascade. This indicates that, though one can imagine that hackers could use multiple shortened URLs and use different redirection chains to evade detection and to be resilient to the blacklisting of particular URLs, they seem to rarely do so. Instead, we find that, within each cascade, hackers typically use the same `Posted-URL` to ultimately redirect users to the target landing page for that cascade.

On the other hand, `Text` and `Resolved-Domain` cascades are consistently different from other cascade definitions. The ARI scores for these definitions are low when compared with the cascades obtained with any other definition. Finally, in contrast to all of these cases, `Text w/ P-URL` cascades exhibit significant variance across different months in their similarity to cascades identified in other ways. In October and November, `Text w/ P-URL` cascades show high similarity to `Text` cascades, while they have high similarity scores with URL-based cascades in the other three months.

3.2.6 Summary

Our analysis suggests that, given that the landing pages corresponding to malicious posts may often be unavailable, the URLs included in posts serve as an effective proxy for identification of cascades. In contrast, hackers often vary text messages across posts in the same cascade. Therefore, even though text included in malicious posts has been proved to be useful in detection of socware [103], identifying cascades by grouping posts with identical text messages can result in skewed findings. For example, in Section 3.2.3, we saw that `Text` and `Text w/ P-URL` cascades exhibit intensive activities on some days in our observation period. However, these turned out to be false alarms resulting from hackers varying the

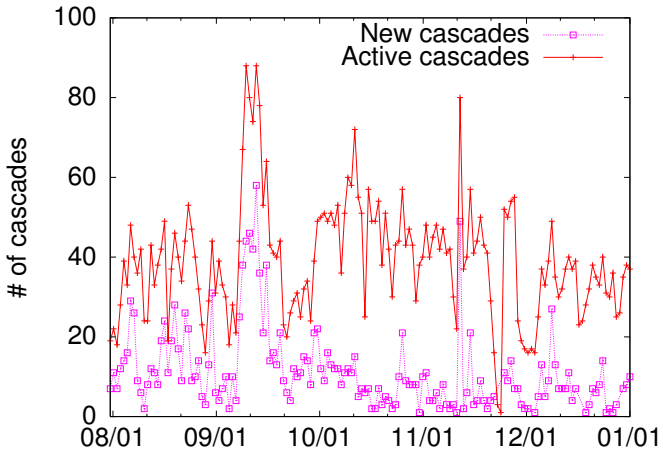


Figure 3.6: Number of active/new cascades over time.

text message within the same cascade.

For the remainder of this paper, we consider cascades obtained by grouping posts with identical posted URLs.

3.3 Cascade Statistics

Having established how to identify cascades, we seek to understand typical characteristics of socware cascades in our dataset: how many cascades are active on any given day? how long do cascades typically last? how are cascades spread across users? While our findings may not be representative for all of Facebook, our analysis here highlights several significant takeaways for anti-socware efforts on Facebook. We present the results from our spatio-temporal analysis of cascades here, and defer a discussion of the takeaways for Section 3.5.

Dataset	cascades with (Active Days = Duration)		cascades with (Active Days < Duration)	
	#	%	#	%
Sep	151	64.26%	84	35.74%
Oct	26	25.00%	78	75.00%
Nov	14	13.46%	90	86.54%

Table 3.5: Comparison of active days and duration for cascades that last longer than a day.

3.3.1 Temporal properties of cascades

We begin our spatio-temporal study of cascades by analyzing the distribution of the number of active cascades over time. Figure 3.6 shows that, on any given day, there are typically around 40 socware cascades active in our dataset. On some days, the number of active cascades goes up to as high as 80. Further, we see that typically less than 10 new cascades begin on any given day. Thus, this seems to indicate that, at any point in time, most active cascades are ones that have been active for a while.

To confirm the presence of long-lasting cascades, we next examine the durations of cascades. To avoid edge effects, we perform this analysis as follows. We consider our *D-Sep*, *D-Oct*, and *D-Nov* datasets, and in each case, we consider the cascades active at some point during the respective month. We then compute the duration for a cascade found in a particular month's dataset by examining the posts for that cascade across the previous month, the month considered, and the subsequent month. As a result, a cascade's duration can be at most 3 months in this analysis.

Most cascades are short-lived. For each of September, October, and November 2011, Figure 3.8 shows the distribution of duration for cascades active during that month.

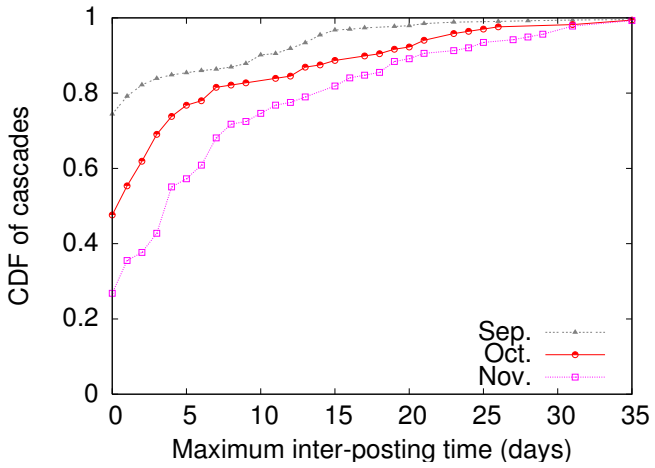


Figure 3.7: Distribution of maximum inter posting times for cascades observed in Sep., Oct., and Nov. 2011.

In all three months, we find that most cascades are short-lived; 60% of cascades last shorter than 1 week. However, there is a long tail of long-lived cascades in all three months; 5%–22% of cascades last longer than a month. It is due to these few long-lived cascades that we previously observed the number of new cascades on any given day to be fewer than the total number of cascades active on that day.

Since MyPageKeeper monitors posts on the walls of a subset of Facebook users, we are potentially under-estimating cascade durations. To evaluate the robustness of our inferred durations, we examine the cascade durations that we would have inferred if MyPageKeeper had half the users that it does have. From the users selected in our *D-Sep* dataset, we pick 10 different randomly chosen subsets with half the number of users. For every cascade in the *D-Sep* dataset, we compute its duration across the posts in each randomly chosen subset and compute the average duration across subsets. Figure 3.9 compares the distribution of cascade durations on our original dataset with that obtained using half

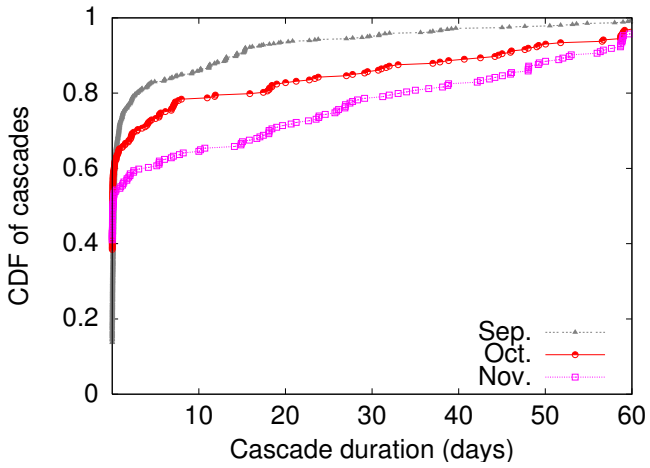


Figure 3.8: Cascade duration for cascades observed in Sep., Oct., and Nov. 2011.

the users as described above. We see that the distributions would largely be the same even if MyPageKeeper had half as many users.

Cascades are becoming less active and lasting longer. Figure 3.8 also shows that newer cascades appear to be lasting longer than older ones. The long tail of long-lived cascades increases significantly from September to November.

To understand the cause for the increasing cascade lifetimes, we examine the activity of cascades in these three months. For each cascade, we count the number of days during its lifetime when malicious posts belonging to this cascade appear in our dataset. We refer to this as the number of active days of a cascade. We consider cascades that last longer than a day and partition them into two sets—1) those which are active every day during their lifetime, and 2) the remaining that have number of active days less than the cascade's duration. Table 3.5 presents the breakdown of cascades into these two partitions for September, October, and November 2011. We see that the fraction of cascades that are active throughout their lifetime significantly reduces from September to October to

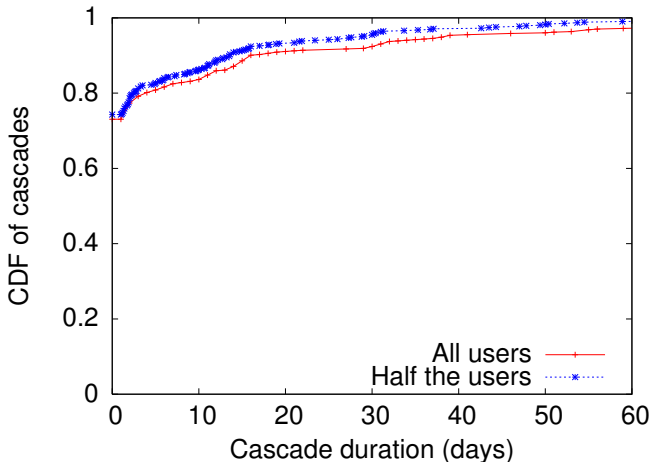


Figure 3.9: Cascade durations on *D-Sep* with full dataset and with half as many users.

November. This seems to indicate that, even within the short time frame of a few months, cascades are evolving to become less active. We speculate that less active cascades are more likely to evade detection by Facebook, and this has resulted in the reduction over time of cascade intensity.

We further confirm the reduction in cascade activity over time by studying the distribution across cascades of the maximum gap in time between successive posts of a cascade. Figure 3.7 shows that the maximum inter-posting time too increases from September to November.

3.3.2 Relationship between users and cascades

Next, we analyze cascades from the perspective of two types of users—1) those who participate in cascades, and 2) those who are exposed to cascades.

Compromised accounts dominate cascades. Users who participate in cascades are those, on whose behalf, malicious posts are made. Figure 3.10 shows the distri-

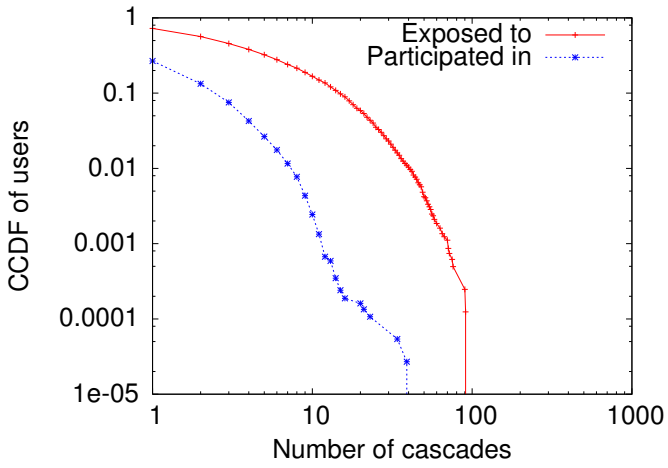


Figure 3.10: Distribution of no. of cascades that users are exposed to or participated in.

bution across such users of the number of cascades in which they participate. We see that 73% of such users were involved in only a single cascade, and out of those for whom we have at least 10 posts in our dataset, more than 99.5% of their posts are benign for over 99% of users. This seems to indicate that a vast majority of users who enable cascades to propagate on Facebook are those whose accounts are compromised, rather than corresponding to accounts controlled by hackers. This is in line with previous findings by Gao et al. [72].

However, we still find 987 users (around 2.65% of all users who participate in cascades) who were involved in more than 5 cascades. Over 95% of these are personal users, 1.3% of them are Facebook pages, and the remaining were deleted when we tried to obtain their information. In addition, we analyze 91 user accounts that participated in over 10 cascades, and find that, for 29% of these accounts, all malicious posts made on behalf of these users were posted by Facebook applications. Furthermore, across these users, on behalf of whom applications are making malicious posts, we found that the average time between when the first and last malicious posts were made by the account is 21 days. We

cascade source	Fraction of cascades
Applications	44.7%
Plugins	32.4%
Manual posting	12.9%
Total	*90.0%*

Table 3.6: Percentage of cascades for which more than 95% of posts originated from a single type of source.

suspect that fraudulent applications installed by these users are making posts with different posted URLs to advertise new cascades over time.

Interestingly, we find that 2.3% of the 15K users subscribed to MyPageKeeper, i.e., roughly 350 users, have participated in cascades. Since users subscribed to MyPageKeeper are conscious of their security on Facebook, the fact that even such users fall prey to cascades highlights the challenges in ensuring that users are not lured by hackers.

The most active friends expose users to cascades. Next, we analyze users who are exposed to cascades, i.e., users who see malicious posts either on their wall or in their news feed. Figure 3.10 shows that 60% of users were exposed to at least one cascade over the course of the five months in our dataset. Among these victims, over 16% were exposed to more than 10 cascades during this period.

To understand if users with more friends are likely to be exposed to more cascades, we group users exposed to similar number of cascades using a bin size of 5. In Figure 3.11, we then plot the 5^{th}, 25^{th}, 50^{th}, 75^{th}, and 95^{th} percentile value within each group for the number of friends. Unlike what one might expect, we see no obvious relationship between

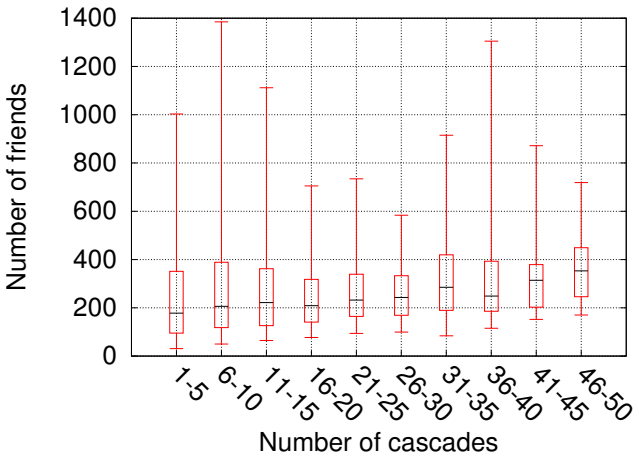


Figure 3.11: No. of friends versus the no. of cascades that a user was exposed to.

the number of friends that a user has and the number of cascades she was exposed to.

However, we do find that users are more likely to be exposed to cascades because of posts made by their most active friends. To show this, we rank every user's friends by the number of posts (both malicious and benign) seen in the user's news feed, and normalize every friend's rank by the total number of friends that the user has. Thus, a friend who makes more posts has rank closer to 0, whereas a friend with fewer posts has a rank closer to 1. As seen in Figure 3.12, over 57% of malicious posts seen in users' news feeds were posted by the top 20% most active friends. Thus, friends who post more actively on Facebook are more likely to expose a user to socware cascades.

3.4 Forensic analysis of socware

Next, we investigate the mechanisms and infrastructure that enable or are used by socware cascades, and the intentions underlying socware.

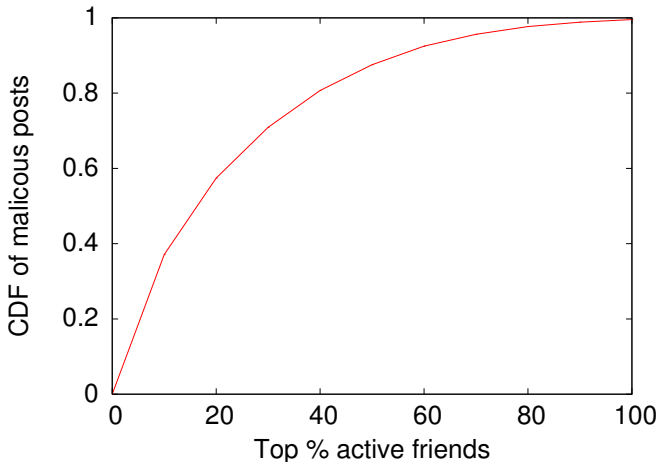


Figure 3.12: Fraction of malicious posts by top x% of friends, ranked by no. of posts.

3.4.1 Provenance of malicious posts

First, we classify malicious posts according to their sources of origin in order to understand what enables these cascades to reach so many users. Posts can be made by any one of the following: (a) a Facebook app, (b) a plugin function connected to a button on a website that the user is visiting, or (c) manually by a user. We refer to these methods as *cascade sources*.

For every post, we examine the *application* field of the post, which contains the *application ID* of the originating application [7]. If the *application* field exists and is empty, it typically means that the post was made by a plugin [5]. If the *application* field is missing altogether, then it indicates that the post was made manually by a user. Note that, while this enables us to determine the source of every post, our dataset does not enable us to track the propagation of cascades across users; this is because, when a socware post is made

[5]This behavior changed sometime in April 2012, but it was true during the period of our dataset, as we manually verified on a sample of posts. Detecting plugin-based posts will require new techniques in the future.

by a friend of a user subscribed to MyPageKeeper, we do not have access to posts that that friend would see on her news feed.

Most cascades (90%) are enabled by one type of cascade source. Since posts in each cascade might be from multiple sources, we first study, for each cascade source, the percentage of cascades in which that source accounted for over 95% of posts. Table 3.6 shows that, for 90% of cascades, more than 95% of the posts were the result of a single cascade source. A potential explanation is that hackers decide on a single distribution method for each cascade. This makes the following discussion easier, since we can associate each cascade with its dominant source.

Roughly 44% of cascades are enabled by Facebook applications. Further, we also observe from Table 3.6 that applications are the largest contributors, with apps covering 44.7% of cascades. Plugins also account for a significantly high fraction (32.4%) of cascades, illustrating the significant role that "Shares" and "Likes" play in enabling socware to spread. Interestingly, manual posting too accounts for a sizeable fraction (12.9%) of cascades. We discuss each category in more detail below.

A deeper investigation of cascade sources. To dig deeper into the different cascade propagation mechanisms, for each cascade source, we manually studied posts from that source, and read blogs and security articles that discuss the source. In addition, we inspected the landing web pages of several cascades. Below, we discuss each category separately and show real examples for each, while Table 3.7 provides an overview of our findings.

a. Applications. Applications enable 44.7% of cascades. According to Facebook's policy, applications require user authorization to post on the user's behalf. Once

Category	Distribution methods
Applications	User installs a Facebook application and the application posts on user's behalf on his wall or on his friends' walls
Plugins	User posts on his wall or on his friends' walls by clicking the Like/Share button from a website
	User clicks on an image or video on a malicious website, where the hacker "hijacks" the user's click to share the link to the website as a status update on behalf of the user
	User gives his Facebook personal upload email address to hackers, who send email to this address to update user's status
Manual posting	Spam posted on the wall of a Facebook page is received in the news feed of a user who follows that page
	User copies specialized javascript snippets designed by hackers into the browser's address bar, thus enabling the hackers to post on the user's wall by utilizing the current session information in the browser.

Table 3.7: Distribution techniques for each cascade source.

an application has been granted permission, it can post whatever and whenever it wants on the user's wall, until the user changes his password or uninstalls the application. A representative example of such a cascade tries to lure people into installing an application by offering to: "Check if a friend has deleted you". We elaborate on the role of applications in enabling socware cascades in Section 3.4.2.

b. Plugins. Plugins are the main source for 32.4% of cascades. In this case, users are lured into visiting a website, where hackers get them to share the link of the website on their wall using the *Share, Like, Recommend* functionalities, which are described as "social plugins" by Facebook [6]. There are three predominant methods that hackers use here: (a) they convince users to knowingly share the website's link on their wall, (b) they trick users

[6]To clarify, there are functions with the same names (Share and Like) which operate on Facebook profiles, but we are not discussing those here, as they do not lead to posts in this category.

to click in such a manner that enables sharing without them realizing it, and (c) they use the Facebook Personal Upload Email, which we discuss below. Note that, since posts made with any of these three methods are indistinguishable in our dataset, we cannot provide a breakdown among the three cases.

In the first case, a website explicitly asks users to click on the *Share* button, often as a condition for the user to get some benefit, such as allowing the user to see an interesting video, e.g., in a cascade which was spreading a photographer's video posts, users are asked to "Click Jaa twice to confirm" that they are older than 18 years; "Jaa" means *Share* in Finnish.

In the second case, websites (which typically contain videos or pictures) use *click hijacking* to post messages on users' profiles without their knowledge or consent. A click-hijacking example seen in our dataset involves a cascade which was advertising the website `http://birthpostions.blogspot.com/` (note the missing "i" in "postions"). When a user clicks the play button on the video at the site, it automatically posts the link on the user's wall. This cascade managed to distribute $95,556$ posts, as per Facebook's statistics.

In the third case, hackers trick users into giving them their *Facebook Personal Upload Emails*. The use of these email addresses is to let users update their status or upload pictures to their profiles by simply sending an email to this address. Hackers, of course, found this as a way to post on users' walls [7]. Once a hacker has this email address for a user, he can send emails that include a malicious URL, which correspond to a status update. This status update appears as a post in the news feed of all the user's friends.

How are the hackers getting these emails? One way, of course, is via malicious

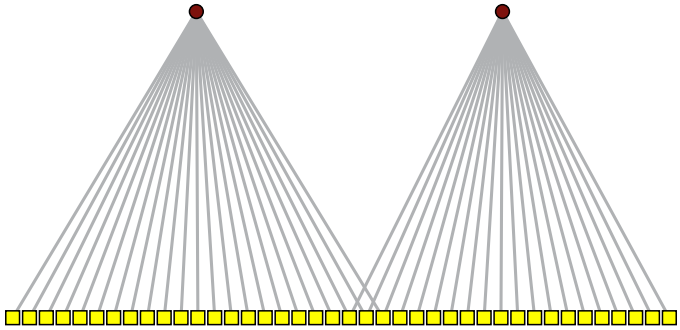
[7]Facebook has removed this email from their mobile website, but the email addresses still work at the time this paper is written.

cascades, which indicates the self-perpetuating nature of socware. In our dataset, we observed cascades that lured people into completing several steps in order to get a fake reward. For example, in the "free Facebook T-shirt" cascade we mentioned above, the fourth step asks users to share their Facebook Personal Upload Email addresses. In fact, hackers have created a YouTube video to show people how to find their email addresses.

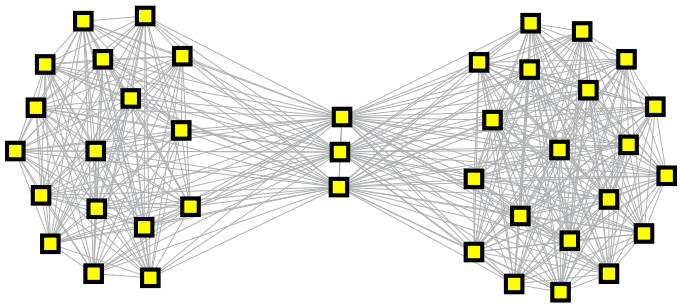
c. Manual posting. 12.9% of cascades are enabled by manual posting as their main source. The posts in these cascades have no application field, and according to the Facebook convention, these posts were posted by users without any help from applications or Facebook's social plugins installed on websites. We identified two large categories of such posts.

First, administrators and owners of Facebook pages (profiles of organizations) can post malicious links on the wall of that profile, possibly unknowingly. As a result, these posts spread to the users who "follow" that profile. For example, the administrator of a page called "I Love My Kids" posted a link to `www.FindJobsForMoms.com`, a fake work-at-home web page. These messages appeared on the news feeds of the users in our dataset that were following that profile.

A second category of manual posts consists of users who blindly follow the instructions on web pages in order to get free stuff. For example, a web page promised "free Facebook T-shirts" and asked users to complete 5 steps. The first step is "Copy And Post Below Message 10 Times on DIFFERENT GROUPS (My Groups) In Facebook."



(a) App-Campaign graph



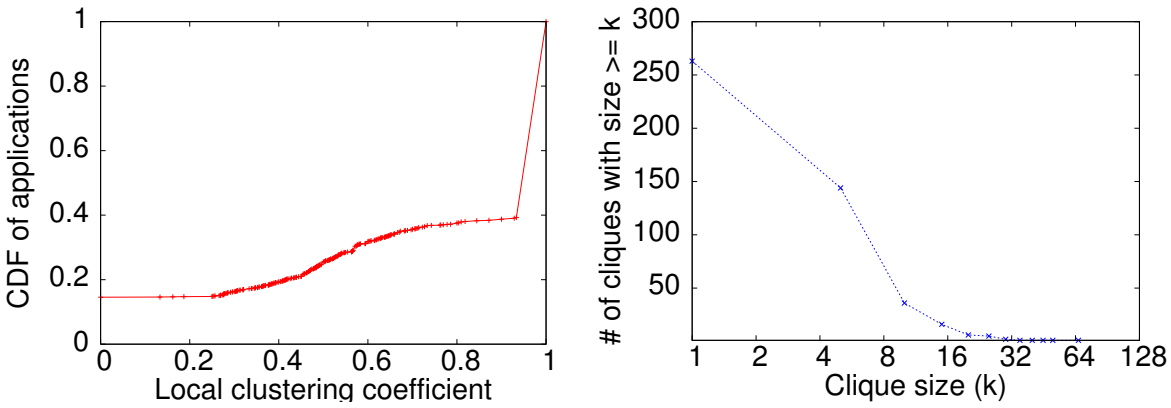
(b) App graph

Figure 3.13: Sample of the real App-Campaign graph showing two large groups of applications enabling two cascades and the corresponding pair of maximal cliques in the App graph. Yellow squares are applications and reda circles are cascades.

3.4.2 Ecosystem of colluding applications

Intrigued by the observation that Facebook apps significantly contribute to enabling socware cascades, we next investigate the relationships between such malicious applications.

The first natural question is: Do these applications collaborate? We consider two graphs. First, we consider a bipartite App-Campaign graph, whose nodes are applications (as defined by applications IDs) and cascades. An edge between an application and a



(a) Distribution of local clustering coefficient.

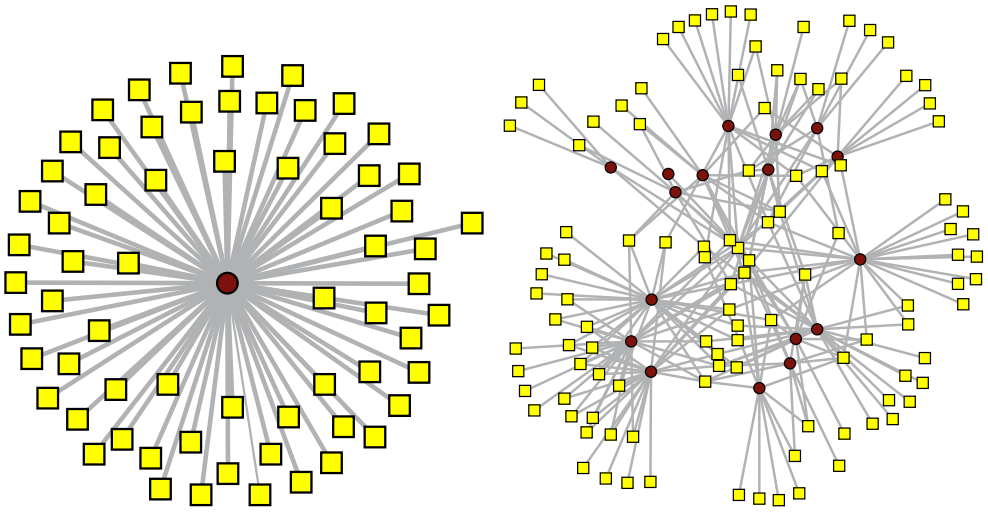
(b) Lower bound on no. of cliques with size >= k.

Figure 3.14: Properties of App graph.

cascade exists if the application has enabled the cascade, i.e., some posts in the cascade were made by the application. Based on the App-Campaign graph, we create the App graph, whose nodes are applications and an edge between nodes exists iff the applications have enabled one or more common cascades; an edge exists between applications A and B if there exists a cascade in which some posts were made by A and some other posts by B. The App graph from our dataset has $1,670$ nodes and $11,612$ edges, with 220 connected components. Figure 3.13 shows a real sampled instance of the transformation between the two types of graphs.

We quantify the collaborative nature and the structure of application interactions as follows.

a. Malicious applications are collaborating: Roughly 60% of applications have a fully connected neighborhood. Figure 3.14a shows the distribution of local clustering coefficient across nodes in the App graph. We can see that over 60% of nodes have their local clustering coefficient equal to 1. This means that 60% of applications



(a) 66 apps support one cascade.

(b) 17 apps support several cascades.

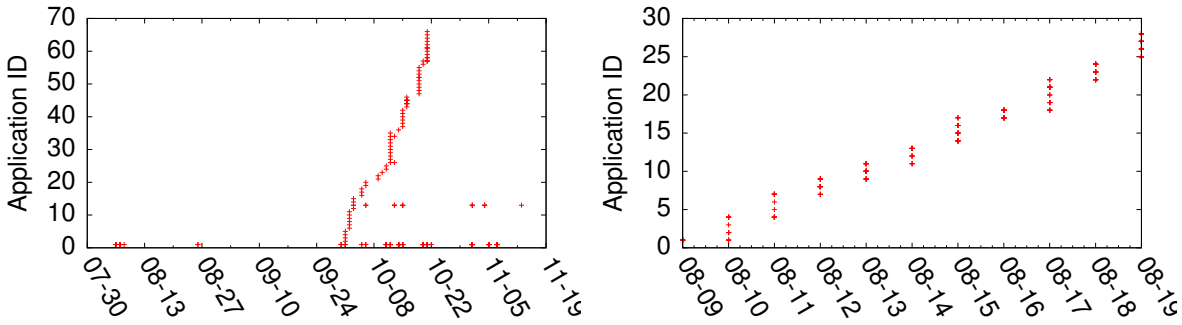
Figure 3.15: Sample of collaboration between applications supporting cascades in the App-Campaign graph.

are such that their one-hop neighbors form cliques, which occurs only if every pair of them shared at least one cascade. Note that 70% of the nodes with local clustering coefficient equal to 1 have degree greater than 5.

b. There are at least 144 cliques of collaborating applications with more than 5 members. We further use a heuristic method to find a lower bound on the number of cliques that are larger than a threshold k [8] In Figure 3.14b, we show the number of cliques at least as large as the x-axis value. We see that there are 144 cliques with size larger than 5, with the largest clique having 66 nodes. Figure 3.15 shows two real samples of colluding applications (the yellow squares) that form a clique in the App graph.

c. Applications collaborate in a structured manner. To further understand

[8]Finding cliques larger than a certain threshold is NP-hard. Our approach hence shows the lower bound of the number of cliques larger than a certain threshold. The specifics of our heuristic are omitted due to space limitations, but the primary takeaway is the existence of many large cliques.



(a) Cascade 1

(b) Cascade 2

Figure 3.16: How apps support the same cascade over time. Y-axis represents unique apps and x-axis is time in days. A point shows that the app propagated posts for the cascade on that day.

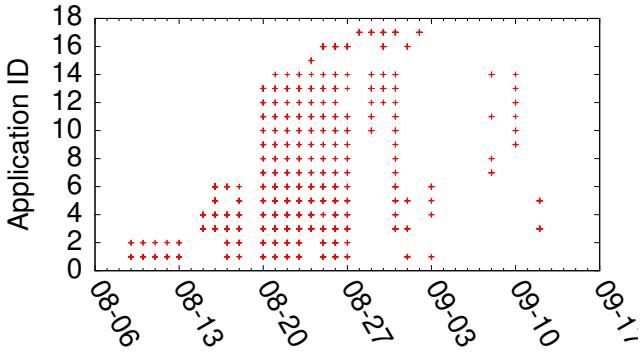


Figure 3.17: The active days of a clique of applications.

the behavior of applications with respect to cascades, in Figure 3.16, we show applications that are supporting a single cascade over time, with a different single cascade per plot. The y-axis represents unique applications in the order when they first appeared supporting the cascade, and the x-axis is time in days. In both cases, there seems to exist a structure

that implies an engineered process in the number of active applications and the observed duration during which an app supports the cascade. This structure is more pronounced in Figure 3.16b, where each application seems to be active for two consecutive days for the cascade in question.

d. The behavior of an application group over time exhibits synchronized activity. Finally, we study how a group of applications is active over time across all the cascades that the group supports. In Figure 3.17, we have identified a collaborating group of 17 applications, and we plot for each application (y-axis) whether it is active on a particular day (x-axis), independently of cascades. We see that the majority of the applications in the group are active during the week 8/20 to 8/27. Note that we show all the activity of this group in our trace; there were no posts from these applications outsides the dates shown.

3.4.3 Intentions of socware websites

Next, we examine the intentions underlying each cascade by examining the landing web pages that posts in the cascade lead to. We identify $1,247$ such unique landing pages in our dataset.

Given the large number of URLs, we cluster them into groups before we investigate them manually. To cluster URLs, we calculate the similarity between every pair of HTML files (without HTML tags) based on the text shingling algorithm [58]. We get 815 clusters, with 40 of them containing more than 2 URLs. We manually check the intention of the cascades in these 40 clusters, which correspond to 691 URLs. We identify five different intentions: *Survey, Phishing, Phishing (as Facebook apps), Advertising Websites,* and *Unknown.* Note that some of these intentions can co-exist; in other words, a cascade

Intention	#/% of cascades	
Phishing and Survey	191	27.64%
Phishing (as apps) and Survey	123	17.8%
Phishing	92	13.32%
Survey	22	3.19%
Advertising Websites	163	23.59%
Unknown	100	14.47%

Table 3.8: Intentions of socware websites

may have two or more of these intentions. We show the distribution among intentions and combination of intentions in Table 3.8.

Most socware cascades do phishing and also benefit from asking users to do a survey. In Table 3.8, we see that 45.44% of cascades want to collect users' personal information (phishing) and benefit from getting completed surveys at the same time. Hackers either create web pages and ask users to complete a form in order to receive fake rewards, or ask users to install malicious applications, which obtain access to information from their Facebook profile. The lure for users to finish surveys are fake rewards, like free iPads or gift cards. After the survey, users are asked for their contact/shipping information in order to ship fake rewards to users. An example we observed was titled "Check if a friend has deleted you". It shows up as a Facebook application which asks users to authorize them to access users' basic information from their profile. After that, users need to finish a survey in order to see the results.

The second largest category is *Advertising Websites*. These websites are usually scam websites or websites that simply want to increase their traffic and popularity. For example, people keep sharing a well-known fake work-at-home website because they believe

that the website is real. Some cascades lure people to websites by promising shocking or funny videos or pictures. However, users need to either "Like" or "Share" this web page with their friends first in order to watch the videos. As mentioned previously, some websites also use click-hijacking to advertise their websites. Facebook has however recently put in measures to prevent click-hijacking; it explicitly requests the user's permission before posting on the user's wall.

The "Unknown" category contains websites which we were not able to access and crawl. The most common issues were: (a) the site had become inaccessible, or (b) the URL was shortened and the shortening service provider had blacklisted it by the time we attempted to resolve the URL.

3.4.4 Cascade hosting infrastructure

Lastly, we analyze the infrastructure that enables cascades. We examined the locations of the servers on which the resolved URLs and landing URLs in our dataset were hosted. We show the names of the top 5 Autonomous Systems that host these spamming websites in Tables 3.9a and 3.9b.

37% of unique Landing URLs are in Facebook's domain. When we look at the landing URLs, 37.07% of cascades are hosted in domains that belong to Facebook. These domains include `facebook.com`, `apps.facebook.com`, and `fb.com`. However, if we look at the resolved URLs, only 4.29% of such URLs are hosted in domains belong to Facebook. These two observations show that hackers use redirection websites between the posted URL and the intended destination in order to protect their websites and applications from blacklists, or to have more flexible infrastructure, or both. The flexibility comes from

the ability to change the redirection destination without having to change the posted URL.

Other observations of interest include: (a) most resolved URLs are hosted by SoftLayer, a large hosting provider, and (b) the majority of landing URLs (85.6%) reside within the United States. Due to space limitations, we cannot expand this discussion further.

3.5 Implications

In this section, we summarize the implications that our analysis presents for identifying cascades and for reducing socware on Facebook. Since our results are based on a subset of all of Facebook's users, these implications are speculative to some extent. However, this is a first attempt to put together a set of take-home messages for those combating socware on Facebook. Since hackers continually evolve, analysis along the lines of what we present in this paper will need to repeated to revisit and revise the takeaways.

a. Defining and identifying cascades. We showed that it is significantly more common for hackers to vary the text message they use across different posts in a cascade, than it is for them to vary the `Posted-URL` included in the posts. Therefore, grouping posts based on the `Posted-URL` is an effective way of identifying cascades in the absence of `Landing-URL` information. While this may change in the future—hackers may begin to more commonly use multiple `Posted-URL`s in a cascade to improve resilience to URL blacklisting—our analysis presents the systematic approach that one can repeat to determine how cascades can be identified.

b. Avenues for improving anti-socware efforts. Our results highlight several promising avenues for anti-socware efforts. Our observation that close to half of the cascades in our dataset were due to malicious Facebook applications suggests that socware on Facebook can be significantly reduced by focusing on the detection of malicious applications. Further, we showed that 37% of socware cascades have their landing URL hosted on Facebook—in the form of pages, events, etc. This suggests that Facebook can reduce socware significantly by better policing of the content on its own platform.

c. Speeding up detection approaches via prioritization. For systems such as MyPageKeeper that identify socware, it may prove to be cost prohibitive to check via crawling every URL posted on Facebook. To maximize the chances of detecting new cascades with minimal resources, such systems can prioritize the checking of certain types of posts over others. Studies like this one can help in this direction. Before our study, a potential strategy for such prioritization could be to preferentially focus on posts seen on the news feeds of users with a large number of friends: the more friends a user has, one may expect the chances of that user receiving socware to be greater. However, our analysis shows that this strategy for prioritizing posts will likely not be effective since the chances of a user being exposed to socware appear to be largely uncorrelated with the number of friends that user has. Instead, we see that a user's posting activity is a better indicator of that user aiding a cascade to spread. Thus, an effective strategy for prioritization is to focus on posts made by highly active users.

On the other hand, it appears that focusing on posts made by users who have previously posted spam would also be largely ineffective. In our study, 73% of users (who participated in any cascade) participated in a single cascade through the entirety of our five

month dataset. This is because most socware on Facebook is posted by compromised user accounts, rather than by accounts owned by hackers, as further corroborated by previous studies [72, 103].

d. Challenges in combating socware. Finally, our results also highlight several challenges in detecting socware on Facebook. First, we showed that hackers appear to be throttling the rate at which cascades grow. This makes it harder to detect the onset of new cascades simply based on the sudden increase in the sharing of a URL, and thus makes it especially important to rely on other features unrelated to the rate of spreading to detect socware.

Second, we saw that reducing socware by detecting and blacklisting malicious applications is made challenging by the fact that hackers use several applications to spread a single cascade. Unless all applications in a cascade are blacklisted, hackers can simply create new applications and continue the spread of the cascade. This calls for the synergistic identification of all applications associated with a cascade.

Lastly, we found that over half of socware cascades were being propagated by users clicking Like/Share on websites outside of Facebook or by users manually posting spam. Users typically do this in the lure of fake rewards such as free products. Therefore, irrespective of measures adopted by Facebook and others to detect socware, user education is a must to eliminate socware from Facebook.

3.6 Conclusions

In this chapter, we presented the first comprehensive study on systematically characterizing socware cascades on Facebook. While most cascades are short, we discovered that hackers seem to be evolving into reducing the activity of cascades in order to evade detection and make cascades last longer. Most interestingly, we showed that a large fraction of socware cascades are supported by Facebook applications that are strategically collaborating with each other in large groups. Lastly, we showed that socware significantly differs from email spam both in terms of the underlying intentions and the tricks used to con users. We believe that these findings will help develop the next generation of anti-socware tools for Facebook.

AS Name	%
Facebook	37.07 %
SoftLayer	10.46 %
Google	8.79 %
Hostway Services	4.52 %
XO Communications	2.85 %
ThePlanet	2.51 %
GoDaddy.com	2.09 %
HOSTING-MEDIA	1.67 %
Bluehost	1.67 %
HostDime	1.59 %

(a) Top 10 Landing URL ASes

AS Name	%
SoftLayer	25.99 %
ThePlanet	10.81 %
Google	8.58 %
Hostway Services	4.54 %
Facebook	4.29 %
American Internet Services	2.89 %
XO Communications	2.89 %
GoDaddy.com	2.81 %
Escatel Network	2.72 %
HOSTING-MEDIA	1.73 %

(b) Top 10 Resolved URL ASes

Country	Percentage
United States	85.64%
Canada	2.00 %
Thailand	1.75 %
Lithuania	1.67 %
Russian Federation	1.59 %

(c) Top 5 countries hosting Landing URLs

Chapter 4

Psychology of socware

In 1949 von Neumann first suggested the existence of self-reproducing computer programs [114]. In 1984 Cohen wrote one of the first computer malware software, which he named *computer virus* [67]. Since then, the connection between computer malware and biological viruses has captivated the imagination of both researchers and the general public [69, 70, 105] and the way by which malware interacts with people and computers has evolved.

Over the last two decades we have seen anti-virus developers locked in an endless arms race against malware developers, much in the way that biological viruses and our immune system are locked in an attack-inoculation struggle. We humans, however, do not rely solely on our immune systems to defend ourselves against biological viruses. Biological diversity in our species is a key ingredient to prevent epidemics, as each of us has different vulnerabilities to the many ways a virus can attack. Having different computer operating systems can be seen as a way to replicate the same biological diversity defense, one to which today's highly specialized computer malware has no way to get around. For in-

stance, smartphone operating system diversity is seen as a deterrent to smartphone malware spread [115].

Today, a new chapter in the evolution of computer malware is being written. Online social network malware, also known as *socware* in the specialized literature [77, 103], unlike email spam – which is mostly sent by spammers directly to intended victims – socware leverages in great part on the "word-of-mouth" propagation of online social networks; every user who falls victim to a socware cascade unknowingly exposes her friends to the same cascade through bogus direct messages or broadcast posts [77]. Moreover, the victim is subject to identity fraud as malware developers impersonate the victim in the online social network to advertise other application or promote various types of businesses.

Through the analysis of posts of over 3.5 million Facebook users collected over seven months, we show that unlike previous generations of computer malware – which are traditionally pieces of highly specialized software that attack specific software vulnerabilities in the host computer – socware targets human incentive mechanisms and cognitive capacity to distinguish between a legitimate request from a friend and a bogus request from an infected friend.

In this chapter we show that most socware posts offer some form of incentive to lure new victims, which we broadly classify as social, financial, or combined (social and financial) incentives. We find that social incentives often target the victim's social capital – increasing one's social capital is known as one of the reasons why people join online social networks [68] – or the victim's social insecurities about his or her social status. Our data shows that while financial incentives seem to be more effective than social incentives (in respect to the distribution of cascade durations), most of the observed malware cascades

offer only social incentives. But one of our most striking findings is that, much like biological viruses, the most effective socware cascades use multiple incentives to get around our diversity in susceptibility to distinct incentives. Through simulations we see that even if the susceptibility of combined incentives is less than that of one specialized incentive, combined incentives still can compromised $70\% - 180\%$ more users than specialized incentives.

4.1 Dataset

Before we describe details of our dataset, we define the terminology we used first.

Definitions. A *cascade* is defined as the set of posts with the same unique URL, that is, we are aggregating posts according to the website they advertise. We have shown in section 3.2 that such aggregation results based on the embedded URL is similar to results based on URLs to which the embedded URLs lead users. A *post text* is a collection of post texts that show small Levenshtein distance [27]. For instance, "Click here and win an iPad2" and "Click here and win an iPhone5" are the considered posts with same post texts.

A single post may contain one or more distinct incentives. For instance, 22.52% of the posts contain both financial and social incentives. These posts are *combined incentive* posts. Otherwise the post has a *single incentive*. Our Mechanical Turk survey is designed to find combined social and financial incentives.

A cascade may have both financial and social incentives in two forms. Either the cascade sends messages with combined social and financial incentives or it uses distinct messages with different single incentives. We denote the later cascade to be *mixed socio-financial* incentivized.

Observation period	July 2011 - April 2012
Number of observed posts with links	111 million
Number of malicious posts	164,304
Number of posts with text content	120,455
Number of cascade	4,389
Number of messages	2,128

Table 4.1: Summary of the dataset.

Datasets Our data was collected through the MyPageKeeper Facebook application [31] from July 2011 to April 2012. Over these ten months, the activity of 3.9 million Facebook users are monitored through the newsfeed of 16,240 MyPageKeeper users[1]. In total, we identified 164,304 malicious posts of which 120,455 contain text content, of which 4,389 are unique URLs. Table 4.1 summarizes our dataset.

In order to classify the incentive of thousands of cascades, we recruit volunteers from Amazon Mechanical turk to classify the incentives of each cascades for us. Based on these labeled cascades, we further build up a classifiers which can classify incentive of malicious posts automatically in the future. In order to train and evaluate the classifier, we separate our dataset by month. In the monthly datasets, each month contains posts belonging to the campaigns which first show in this month in our dataset. For example, in D_Sep, we remove all the posts of campaigns which already show in D_Jul and D_Aug. Moreover, we remove all the posts which we are not sure about their incentives from our datasets. Table 4.2 shows the summary of our monthly datasets.

[1]Facebook selectively shows only 12% of the users' friends updates on users' newsfeed, so we won't see all the update of users' friends.

Datasets	# of campaigns	number of posts
D_{Jul}	142	955
D_{Aug}	344	9123
D_{Sep}	485	15152
D_{Oct}	165	8351
D_{Nov}	149	3091
D_{Dec}	102	4551
D_{Jan}	214	6195
D_{Feb}	645	12887
D_{Mar}	320	16063
D_{Apr}	147	1731

Table 4.2: Summary of the monthly dataset.

4.2 Methodology

We classify the different types of incentives into one of the fourteen different incentives shown as a hierarchical taxonomy tree in Figure 4.1. At the highest level, we divided incentives to be two categories, *financial incentives* and *social incentives*. Under these two main categories, we further divided them into subcategories based on the mechanisms that are used to attract new victims. It is space consuming to list sample message for each sub category. Instead, Table 4.3 lists 3 selected frequent words from each category. In our earlier work, we show that the keywords on Facebook are very different from these of email spam [103].

We then use Amazon's Mechanical Turk platform[2] to recruit volunteers to classify thousands of unique Facebook posts. We call these volunteers *turkers* in the remaining of the study. There are several advantages in the use of Mechanical Turk to classify the

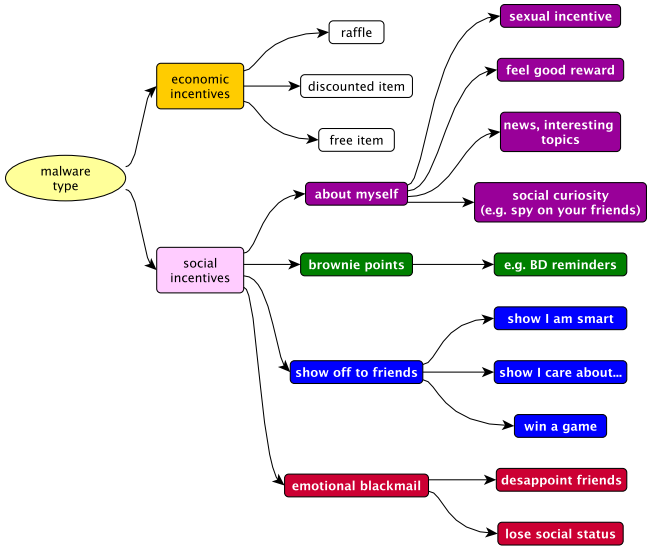


Figure 4.1: Taxonomy of spam incentives.

text messages for us. First, the results actually reflect the feeling when a person read the messages. Second, it can remove the potential bias from collection results only from a few users. In this study, we collect results from 226 turkers in total. Third, we can evaluate our designed incentive categories based on users' feedbacks. If turkers have trouble to assign post text to appropriate categories, it implies that the designed categories are not appropriate.

In section 4.5, we further design a classifier, which is based on these manually classified results, to automatically classify different types of posts into their correct categories.

A non-trivial part of malicious posts do not contain text messages but still propagate through Facebook. In our dataset, there are 31.68% of malicious posts do not contain any text content in the messages. These malicious posts display something that Facebook automatically grabs from the webpage which the link points to and show it as the posts. As a result, when users read the post, they are subject to an incentivized post

Category	**Frequent Words**
Click and win	Free, wow, offer
Play a game	contest, win, click
Discounted item	free, recharge, get
Raffle	cruise, win, holidays
Earn Money	make, moms, year
No financial incentive	you, check, profile
A chance to support something or to show I can about something (e.g. sport team, animals, etc.)	team, vote, favorite
Best my friends in a game	context, win, click
Brownie points	just, got, miscrits
Social curiosity (e.g., track profile visitors, monitor friend activities)	check, friend, deleted
News, interesting topics, etc. (e.g., Bin-Laden death video)	live, news, streaming
Feel-good reward (e.g., video of puppies)	freee, shirt, recharge
Sexual incentive	omg, shows, model
Personalize and make my Facebook page cooler	Facebook, graphic, change
No social incentive	free, wow, offer

Table 4.3: Frequent Words in each category

even though there is no text content in the post. We do not consider these posts in this study. In order to study these the incentives used by these posts, we need to collect the content of webpages pointed by the links. It is hard to collect the necessary information because that these these webpages are usually short-lived[77]. The dynamic nature that the link could point different users to different pages create another difficulty. We left the analysis of the incentives of these kind of posts for future study.

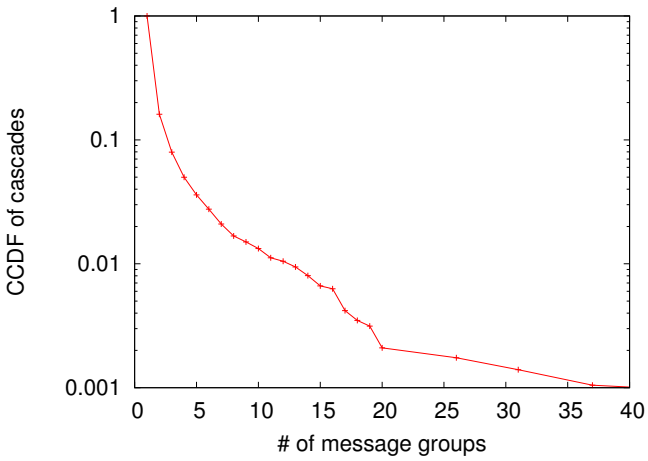


Figure 4.2: CDF of number of text message groups

4.3 Analysis of Facebook spam incentives

In this section we report the observed incentives at the cascade level. We also study these cascades using metrics such as number of infected accounts and duration of cascades.

In total, we have identified posts belonging to 2,863 cascades. In this section we summarize our findings. We start observing that most cascades (84%) maintain the same post text throughout the life of the cascade. There could be several benefits for spammers to use different post texts to promote their cascades. For example, 1) avoiding detection by Facebook. If posts with similar texts spread through Facebook, they will attract attention from Facebook. 2) Attracting different type of users. Different users might be attracted by different social engineering techniques. It could lure different type of users by change post texts with different incentives. However, we only observe 16% of cascades has ever change post text throughout the life of the cascades. It could be because that the cascades are usually short lived[77]. The websites are down before they use different text to promote

their websites. The other possible reason could be that the spammers don't need to change text to avoid detection. Figure 4.2 shows the CCDF of number of post texts across different cascades.

To check if the same post text is shared by multiple cascades, we also study the distribution of number of distinct cascades advertised with the same message. Here, we observe that only 16.1% of post texts advertise more than one cascade. This, combined with our previous result, suggests that most cascades have a unique message.

Interesting, however, we also observe multiple cascades share the same post texts. For example, we found that post texts containing *"Check if a friend has deleted you"* advertise 591 different cascades (URLs). This either shows an uncommon level of sophistication among cascades or shows that the post text is being copied by other spammers due to its effectiveness.

We also observed some cascades used multiple post texts. For example, a cascade, which is advertising *http://SportsJerseyFever.com/NFL*, use 37 different post text. They create several pattern of messages and change the team name in each pattern to attract fans from different teams. Two different patterns are listed bellow.

- Do You Still Love The Steelers? Steelers Steelers Steelers! Got Steelers Fever? Show Your Steelers Fever & Vote For Them As Your Favorite Team! Vote Steelers −− > http://SportsJerseyFever.com/NFL Vote Now - Expires In 15 Minutes!

- FOOTBALL SUNDAY!! I Love My Steelers! Got Steelers Fever? Show Your Steelers Fever &; Vote For Them As Your Favorite Team! Vote Steelers – http://SportsJerseyFever.com/NFL Vote Now - Expires In 15 Minutes!

Table 4.4: Distribution of incentives among cascades:

(a) financial incentives

Incentives	Observed cascades
Click and win	466 (17%)
Play a game	53 (2%)
Discounted item	9 (0%)
Raffle	7 (0%)
Earn Money	77 (3%)
No financial incentive	1941 (71%)
No consensus	48 (2%)
Mixed financial incentives	117 (4%)

(b) Social incentives

Incentives	Observed cascades
A chance to support something or to show I can about something (e.g. sport team, animals, etc.)'	22 (1%)
Best my friends in a game	40 (1%)
Brownie points	6 (0%)
Social curiosity (e.g., track profile visitors, monitor friend activities)	1206 (44%)
News, interesting topics, etc. (e.g., Bin-Laden death video)	105 (4%)
Feel-good reward (e.g., video of puppies)	136 (5%)
Sexual incentive	12 (0%)
Personalize and make my Facebook page cooler	28 (1%)
No social incentive	611 (22%)
No consensus*	404 (15%)
Mixed social incentives	148 (5%)

(a) Only 30% of the cascades use financial incentives while 82% of the cascades use social incentives. The incentives of a cascade are determined by the contents of its post texts. A cascade may use multiple incentives in two ways: either it uses multiple post texts which contain distinct incentives or the post text itself contains combined incentives. An example of a cascade with combined incentives is *"I finished 10th to win an iPad2 in the Jewell Game! Can you beat me?"*. In our dataset, we observed 480 cascades with distinct financial incentive post texts and 1848 cascades with distinct single social incentive post texts. Table 4.4 lists the distribution of incentives among cascades. Moreover, there are 298 cascades with post texts that contain combined/mixed social and financial incentives.

The most prevalent financial incentive is *"Click and win"*, which is present in 17% of all financially incentivized cascades and is also in 37% of all observed post texts with financial incentives. Social incentives, however, are present in 82% of all cascades. Among these, *"Social curiosity"* is the most prevalent accounting for 44% of all cascades. Cascades with this incentive contain post texts such as *"Check if a friend has deleted you"* and *"Check who visited your profile"*. These incentives exploit users' social curiosity, and needless to say, don't deliver the promised functionality.

In order to make the comparison of cascades with different incentive easier, we assign incentives to cascades at the highest level of taxonomy of spam incentives. We divide cascades into three different categories based on their incentives: (a) pure financial incentives, (b) pure social incentives and (c) combined/mixed socio-financial incentives.

(b) On average, there are at least 62% more compromised accounts involved in cascades with combined/mixed socio-financial incentives than with pure financial or social incentives. Moreover, combined/mixed socio-financial cascades

	# of cascades	# of compromised accounts	
		sum	average
Pure financial	482	6323	23.29
Pure social	1848	33828	27.12
Mixed socio-financial	298	8368	44.06

Table 4.5: Statistics of incentives

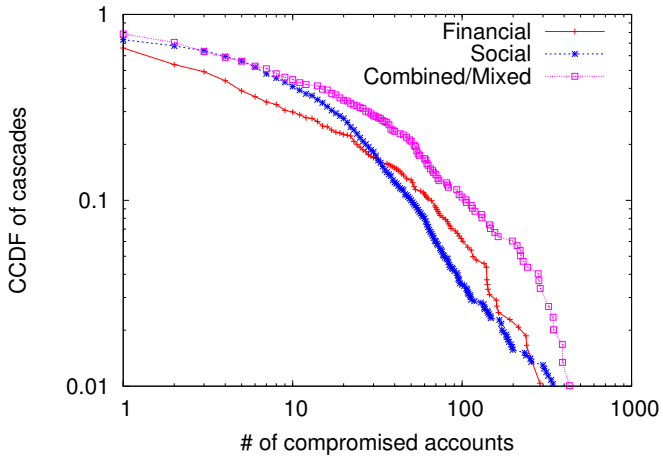


Figure 4.3: Distribution of compromised accounts across cascades

last at least 1.36× **longer than cascades with pure incentives.** We consider a account as a compromised account if she made a malicious post. It has been shown that most of the accounts who make malicious posts on Facebook are compromised accounts [103][72]. In total, cascades with pure social incentives have the most number of involved compromised accounts. However, the number of cascades with pure social incentives is also the largest. It is 3.82 times more than pure financial incentives and 6.18 times more than combined/mixed socio-financial incentives. When we look at the number of compromised accounts involved in different incentives per cascade, combined/mixed socio-financial actually has the most number of compromised accounts involved on average. It has 44.06 compromised accounts

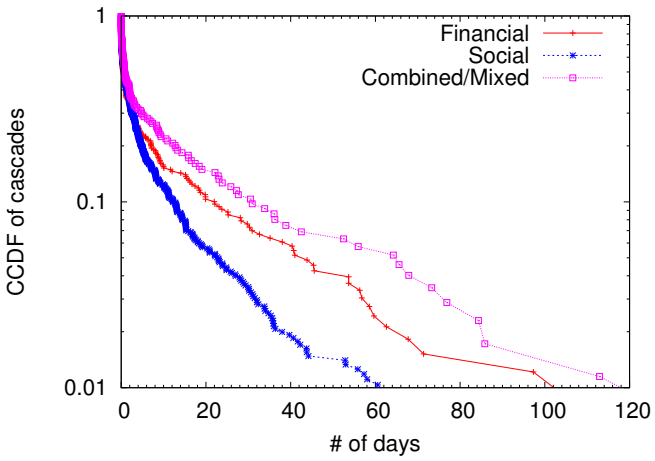


Figure 4.4: Duration of cascades

in average which is around 62.46% more compromised accounts involved in cascades with pure social incentives. Table 4.5 show the statistics of different incentives and Figure 4.3 shows the distribution of number of compromised accounts involved across cascades.

The duration of a cascade is defined as the period of time from the first day until the last day we saw a post originated by a new infected user. Note that we consider a cascade to be alive only while it keeps infecting new users. We don't consider a cascade to be alive if the new posts are posted by existing compromised accounts. Clearly cascades with only one post have no duration. Table 4.6 lists the number of cascades with more than one post for each incentive and the average duration of cascades. We observed that cascades with combined/mixed socio-financial incentives on average last almost 2.5 weeks on average which is $1.36 - 2.75$ times longer than cascade with pure financial or social incentives. Figure 4.4 shows the CCDF of cascade duration across cascades. We also observed several long lived cascades from pure social or financial incentive even though the average cascade duration for pure incentives is much shorter than mixed socio-financial incentives.

	number of cascades	average duration (days)
Pure financial	329	7.54
Pure social	1352	4.76
Combined/Mixed socio-financial	219	17.83

Table 4.6: Duration of cascades with different incentives

(c) Social incentives dominate among all incentives for the whole observation period. After understanding the static characteristics of incentives, we are also interested in the dynamic evolution of incentives. Since spamming activities vary over time, different months may have different number of cascades. It is hard to compare the incentive distribution of each month. As a result, we decide to calculate the portion of cascades with different incentives instead of the absolute number of cascades with different incentives over time. Figure 4.5 shows the fraction of cascades with different incentives observed within each month.

During our 10 month observation period, social incentives are responsible for 63% of active cascades on average. It shows that social incentives always are the top choice when spammers try to advertise their socware. We also observe that there is a sudden spike for combined/mixed incentives in the end of 2012 (October-December). After this period, the portion of cascades with combined/mixed incentives decrease again.

(d) On average, cascades with combined socio-financial incentives show more aggressive posting behavior than cascades with pure incentives.

Different cascades might follow different strategies to advertise themselves. A cascade could spread itself once when it first compromised an new account. On the contrary, a cascade might keep making new posts with its compromised accounts to advertise itself.

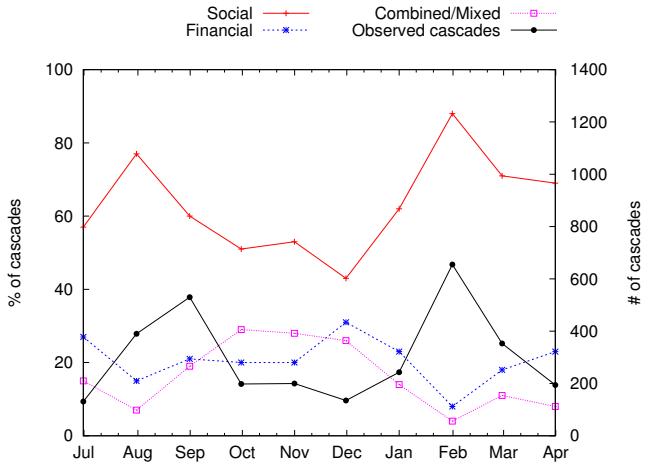


Figure 4.5: Fraction of cascade with incentive i over time

In order to get a better idea of posting behavior of compromised accounts for cascades with different incentives, we design a metric, $P^(i)$, which stands for the number of posts per unique compromised account. It shows aggressiveness of the posting behavior of cascades. Equation (4.1) describes how we calculate the value.

$$p^{(i)} = \frac{\text{total number of posts with incentive i}}{\text{total number of compromised accounts with incentive i}} \tag{4.1}$$

Since a cascade could use different policy with they are using different incentives, we only consider cascades which use the same incentives for the whole spamming duration.

As the results, we remove cascades with mixed incentives from consideration. The number of posts per compromised account for each incentive are listed in Table 4.7. We found that cascades with combined socio-financial incentives show more aggressive posting behavior than cascades with pure incentives. On average, each compromised made 4.14 posts in cascades with combined socio-financial incentives. It is at least 74% more posts per compromised accounts than cascades with pure incentives.

Incentive	# of posts per compromised accounts
Pure financial incentives	2.38
Pure social incentives	1.83
Combined socio-financial incentives	4.14

Table 4.7: Number of compromised account per post

4.4 Simulation

In order to evaluate our observation that **"combined/mixed incentives spread wider than pure incentives"**. We build an simulator in Python to simulate how socware cascades through the network. The detail of the simulation are describe as following:

Mathematical Model

There are more and more researchers who study the diffusion of information recently [60, 74, 93, 96, 123], However, it is still a challenging task to modeling the diffusion in social networks because of the issue of model the underlying process. There is no simple model which could describe the system well. As the goal of our study is to verify our observation which shows that "socware cascades with combined/mixed incentives compromised more users that one with pure incentive", we decide to use a modified SIR(*Susceptible-Infected-Recoved*) model which is the basic but popularly used model for analysis of online social network as our socware propagation model [64, 83, 63]. The other reason we choose SIR instead of SIS is that we believe users will learn from experience. As the result, they have a chance to "recover" from a certain type of socware and become immune to it.

Before we describe the detail of our modified SIR model, we describe the notation of the model first. In the graph, each node (N), which stands for a user in a social network,

has a tendency to a type of incentive (I). Each socware (S) uses one type of incentive to lure users in the network. When a user sees a malicious message, which is sent by a socware with incentive (I_a), it will be infected by the socware with infection probability P_{pure}. The value of P_{pure} equals to P_{pure_ij} where i is the incentive, which the user is susceptible to, and j in the incentive used by the socware. In our simulation, we set $P_{pure\_ij} = x$ if i equals to j; otherwise , $P_{pure\_ij} = 0$

The simulation starts from a single seed in the largest connected component. After a node is infected by a socware (S) with incentive (I), it starts to infect its one-hop healthy neighbors. The infected node will spread the same socware with the same incentive which it is compromised by. The infection probability P to a certain neighbor of the infected node decreases by a dumping factor (α) when the number of time it has failed to compromised this neighbor increases. The equation is shown in Equation 4.2

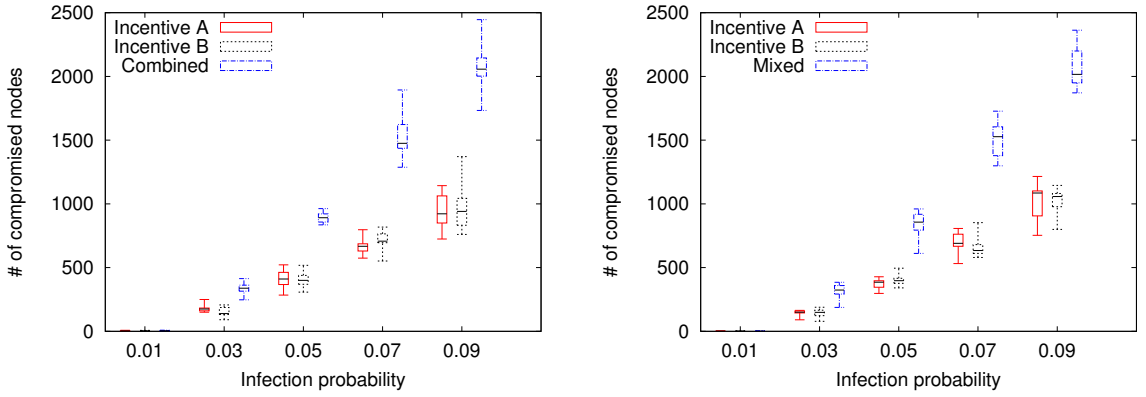
$$P(t) = P_{pure\_ij} \times \alpha^t, \alpha \sim (0, 1] \tag{4.2}$$

where t is the number of times it has failed to compromised this neighbor.

The goal of our simulation is to compare the performance of socware with pure incentive and socware with combined/mixed incentives. For simplicity, a socware with combined/mixed incentives only combines/mixes 2 different incentives (A and B). We define the infection probability for socware with combined as $P_{combined}$ which is calculated as Equation 4.3.

$$P_{combined} = \frac{P_{pure\_iA} + P_{pure\_iB}}{2}, \text{where } i \text{ is the incentive the node is susceptible to.} \tag{4.3}$$

Since socware with mixed incentives sent messages with different type for the whole cascade duration, we define the infection probability P_{mixed} to be either P_{pure_iA} or P_{pure_iB},



(a) Performance of socware with combined incentives (b) performance of socware with mixed incentives

Figure 4.6: Performance of socware with Combined/mixed incentives

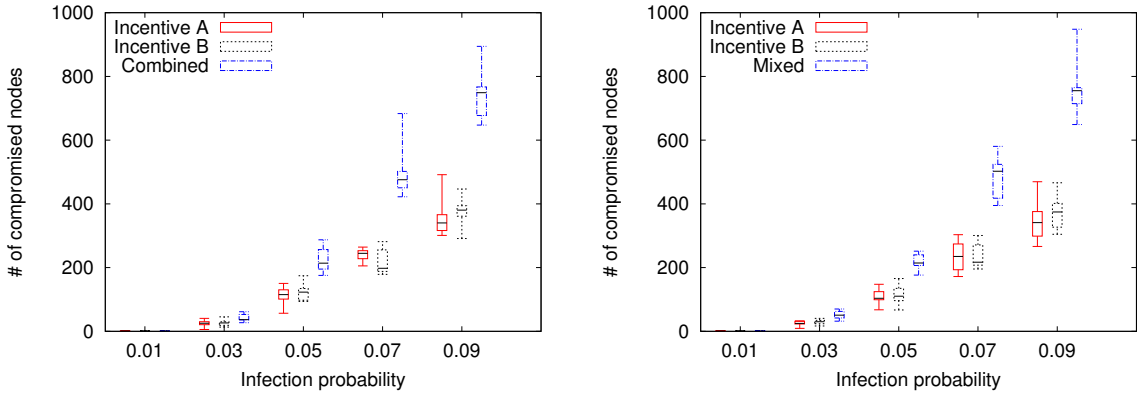
which depends on the type of message it sends at that time.

The Network Data

We employ *Enron* communication network, which consist of 36,692 nodes and 18,381 edges. The largest connected component has 33,696 nodes. The network was built from a trace of email exchange records in Enron Inc. The Enron network is a widely used social network. It is small enough such that we can have many simulation runs (as the code is not optimized for performance yet). For more information about the characteristic of this dataset, please read [84]. The total simulation length is 256 unit times and the results is the average of 3,000 cascades.

At first, we want to examine if socware with combined/mixed incentives spread wider than socware with pure incentive under certain scenarios. The performance metric we use to compare two socware is the total number of compromised accounts.

Scenario 1: The first scenario we want to simulate is that a Facebook user doesn't notice that her account is compromised by an socware or she doesn't know how to uninstall



(a) Performance of socware with combined incentives

(b) Performance of socware with mixed incentives

Figure 4.7: Performance of socware with Combined/mixed incentives

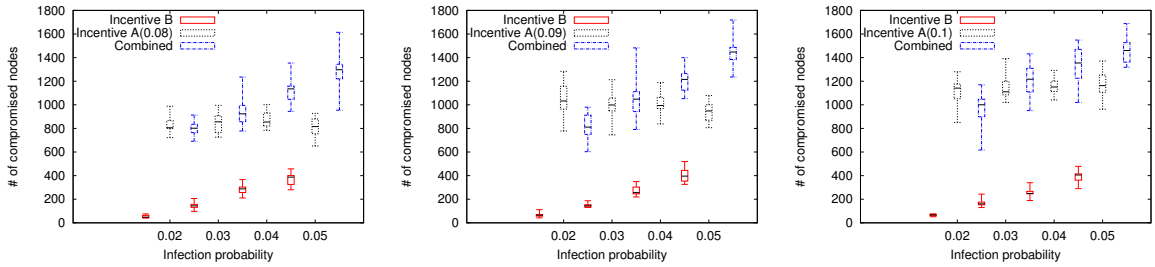
the socware from her account. The socware posts something on her wall so that all her friends will see the malicious posts. The infection rate is high at the first time the socware post something on her timeline. After that, the infection rate drop quickly. In order to simulate this scenario, we set two socware to use pure incentives with the same infection probability. We also set the recovery probability to be 10^{-3} and dumping factor (σ) of infection probability to be 0.5 for the socware. The results are shown in Figure 4.6.

From Figure 4.6, we can see that both combined and mixed incentives compromised more users than pure incentives. As the infection probability of pure incentive increase, socware with combined/mixed incentives outperform socware with pure incentives more. The propagation method of socware, which is word-of-mouth model, could be the reason that makes this effect. In order to infect more users in the network, a infected node has to infect his one hop neighborhood first. Otherwise, the infection stops. With the help of combining multiple incentives into socware, it increases the probability that a compromised node can infect some of nodes around her. As the results, the infection keeps spreading.

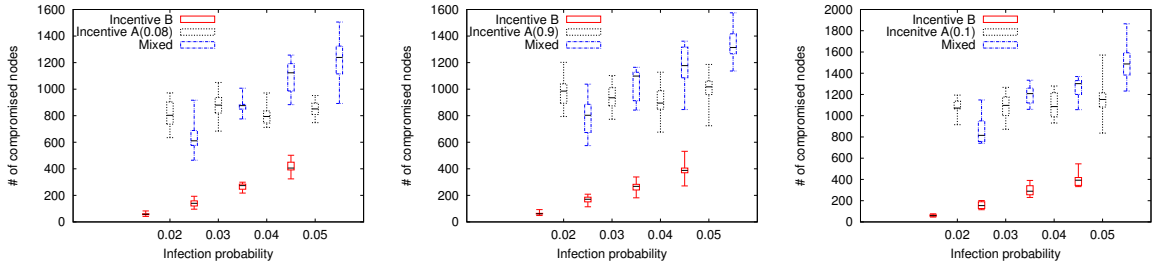
Scenario 2: The second scenario we want to simulate is that an infected Facebook user keeps trying to infect her friends. The infection probability that she can successfully compromises her friends is not related to the number of times she has failed. However, this time, the infected users could recover after they first try to infected their friends. The reason that we make infected users could recover only after their first try to infect their friends is that most of the socware automatically make a post on behave of infected users right after users install socware. In order to simulate this scenario, we set the infection probability with a dumping vector 1 and the recover probability is set to be 0.5. The simulation results are shown in Figure 4.7 which present the same trend as what we observed in the first scenario, "Combined/mixed incentives outperform pure incentives."

Through our analysis, we find some successful socware cascades with pure incentives. Socware contain post text "Check if you friend has deleted you" is a good example. After showing that "Socware with combined/mixed incentives do outperform socware with pure incentives", we are interested in answering the following question: is it worth for socware with highly infectious incentives to combine/mix another less infectious incentive?

We run our simulation with one high infectious incentive and one low infectious incentive on both of the scenarios we describe above. The results are shown in Figure 4.8 and Figure 4.9. From Figure 4.8 and Figure 4.9, they show that combining/mixing with another incentive with a certain level of infectiousness can help the highly infectious incentive compromises more users than it does alone by itself.



(a) Combined a highly infectious incentive with a low infectious incentive



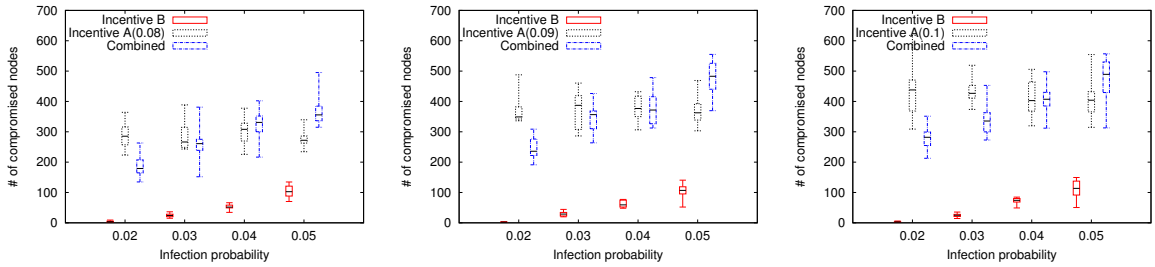
(b) Mixed a high infectious incentive with a low infectious incentive

Figure 4.8: Performance of socware with combined/mixed high/low infectious incentives (Scenario 1)

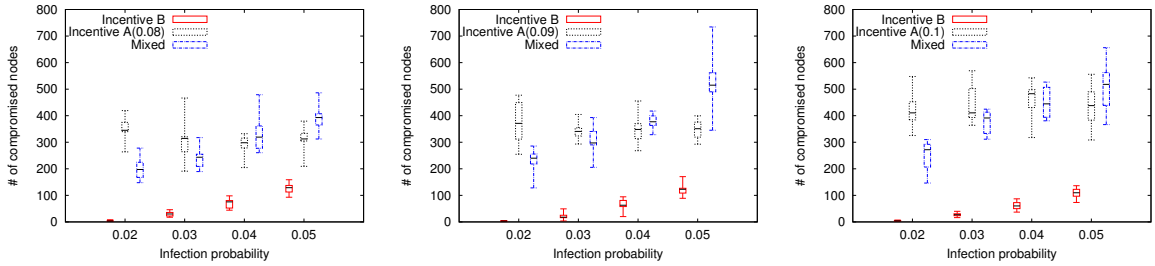
4.5 Automatic classification of incentives

In this study, we recruit Turkers from Amazon turk to classify the incentives of post for us. However, relying on manually classification of posts has several drawbacks, e.g. time consuming and financial costs. Therefore, we build a classifier, which automatically classifies the incentives of the posts, based on modern text classification techniques.

We use bag-of-unigram and bag-of-bigrams(pair of consecutive words) together to represent each post. It has been shown that combining these two can achieve higher accuracy than using bag-of-unigram alone in short text classification [125]. After representing each posts by bag-of-bigrams, the problem turns to be how to assign value to each feature.



(a) Combined a high infectious incentive with a low infectious incentive



(b) Mixed a high infectious incentive with a low infectious incentive

Figure 4.9: Performance of socware with combined/mixed high/low infectious incentives (Scenario 2)

In this study, we assign value to each feature by using Term Frequency Inverse Document Frequency (TF-IDF). More specifically, the value of a word is the number of times this word appears in the posts weighted by the Inverse Document Frequency. The Inverse Document Frequency is calculated by dividing the number of posts containing the words divided by total number of posts and then taking the logarithm of the quotient. We used LibShortText library[126] for SVM based classifications.

The performance metrics we used to evaluate our classifier are precision, recall and F1-score. We also compared our classifier with baseline classifier. The baseline classifier follows the intuition that there is a higher probability that they are using the same incentives if two posts contain similar post texts, thus, the baseline classifier labels the post in the

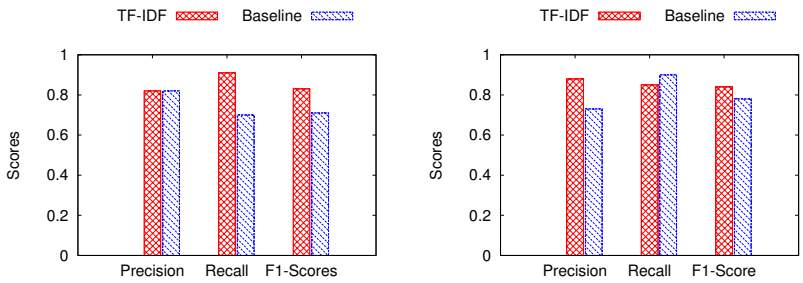
testing month with the same incentive as the label of the most similar post in the training month. We used Levenshtein distances as similarity metric when we compared two posts.

At first, we try to evaluate our classifier with limited training data. We trained the classifiers with the month before the testing month, and classify the incentives of posts in the testing month dataset. For example, we trained our classifier with july dataset and predict the incentives of posts in august dataset. We want to mention it again that we remove all the cascades which shown before from each monthly dataset. It means that the classifier classifies the posts of new campaigns only. Figure 4.10 shows the performance of our classifier and baseline classifier.

Our classifier correctly classified whether a post contains financial/social incentives with high F1-scores. Here, we show the classification results at the highest level of taxonomy tree. We called it *coarse granularity*. At this level, the classifier decides whether a post contains financial/social incentives or not. Our classifier correctly classified the incentive of posts with high F1-scores, which is 0.83 and 0.88 respectively on average . When we compared our classifier with baseline classifier, our classifier outperformed $3\% - 17\%$ higher f1-scores than baseline classifier. Moreover, the baseline classifier needed an average of 1.06 hours to finish the classification. Our classifier needed only 1.26 minutes.

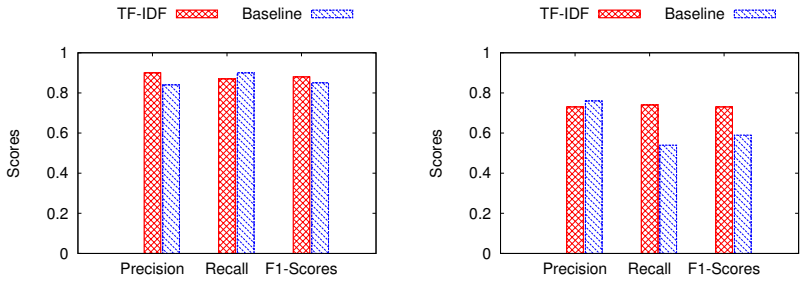
We also evaluate our classifier by using distinct post texts instead of posts. The same post texts could be used in multiple posts. Classifying correctly a post text which is used by a lot of posts might hide the drawback of the classifiers. The classification results of distinct post texts are similar to posts. It shows that the situations mentioned above don't cause the good performance of our classifier.

Social incentives are harder to be classified, thus, they are more dan-



(a) Does a post contain financial incentives?

(b) Does a post not contain financial incentives?

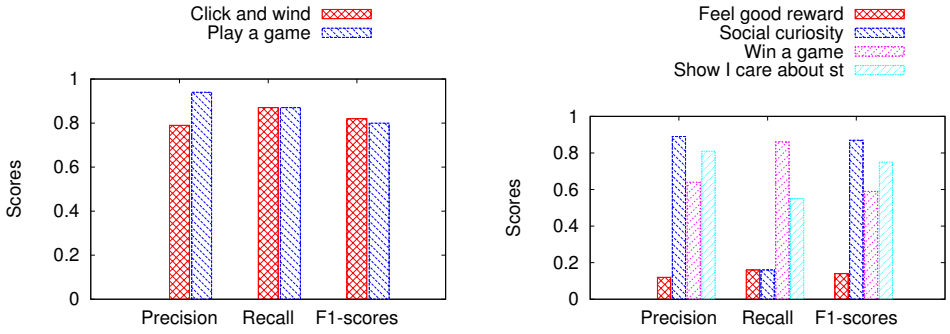


(c) Does a post contain social incentives?

(d) Does a post not contain social incentives?

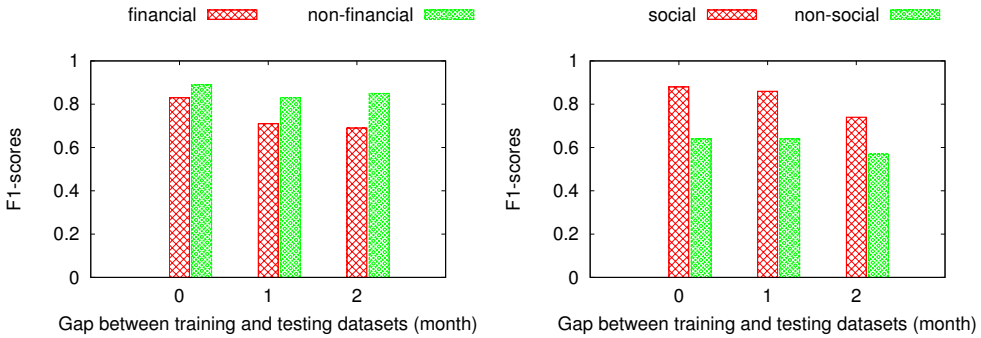
Figure 4.10: Performance of classifiers (coarse granularity)

gerous. After showing that our classifier can correctly classify incentives of posts in coarse granularity, we evaluated the performance of the classifier in classifying incentives of posts in fine granularity, which is at the lowest level of taxonomy tree. We trained the classifier with the first four month datasets and classified the incentives of posts in the remaining three months. The reason that we didn't examine the performance of classifiers which were trained by one month only is that some incentives don't show in all the months. If a subcategory of incentives doesn't appear in the training month, the classifier cannot classify the incentive of posts correctly. Figure 4.11 shows the classification results of each subcategory. We only show classification results of subcategories which contains more than 200 posts in



(a) Subcategories of financial incentives (b) Subcategories of social incentives

Figure 4.11: Performance of classifier (fine granularity)



(a) Financial incentives (b) Social incentives

Figure 4.12: Accuracy of classifier with different time gaps between training/testing datasets

the testing or training datasets.

Our classifier classified subcategories of financial incentives with very high f1-scores. However, the performance dropped a lot in classifying subcategories of social incentives. We conjecture the possible reason is that a post with social incentives usually contains more complicated text content. There are a lot of ways to express social incentives. However, for financial incentives, there are keywords which are very popular in certain subcategories. As a result, it is harder to classify social incentives than financial incentives, thus it makes social incentives more dangerous.

Our classifier has best performance in classifying the month right after the training month. The performance decreases slightly month after month In order to understand the effects of the time gaps between training and testing datasets, we trained our classifier with one-month data and classified datasets which are x months after the training dataset. The results are shown in Figure.4.12. It shows that we have the best performance when there is no gap between training and testing months. As the gap increases, the performance of classification decrease slightly. We conjecture that spammers used similar keywords to express incentives in posts for their campaigns.

4.6 Conclusions

In this chapter, we investigate the incentives used by socware to lure users into their traps. When email spammers directly spam emails to intended victims, socware relies on "word-of-mouth" propagation of online social network. Every infected user helps socware spread farther by unknowingly exposing their friends to the same socware cascades.

We show that socware target human incentives mechanism and cognitive capacity to distinguish between a legitimate request from a friend and a bogus request from an infected friend. After broadly classifying the incentive used by socware to social, financial, or combined (social and financial) incentives, we identify that social incentives are the most popular incentives. Moreover, we found that socware behaves like biological viruses, which use multiple incentives to get around users' diversity in susceptibility to distinct incentives. Through simulations we see that even combining a less infectious incentive with a infectious incentive, combined incentives still outperform one specialized incentive.

Chapter 5

Comparisons between socware and email spam

Since its appearance in 1978, the volume of spam email has increased rapidly, with recent estimates identifying 107-billion spam email message globally per day. It shows that it is still a serious problem today. However, as shown in the report form Symantec [49], that spammers have shift their attentions from spam email to online social networks. Moreover, Socware spread very fast in online social networks because of users unfamiliarity of them. For example, there is a Facebook spam that compromised 5 million users with 48 hours.

After uncovering the characteristic of socware cascades, in this chapter, we would like to compare these two generation of spamming behavior, "spam email" and "socware".

In order to compare spam email with socware collected by MyPageKeeper, we built a spam email dataset by ourselves. There are several reasons that we built the spam email dataset by ourselves. First, spam email could contain personal information. Because

of privacy concerns, people usually don't share the spam email datasets. Second, we need the dataset to be new. Spammers keep evolving and changing their techniques to maintain their business, so we don't want to compare socware with an old dataset. Furthermore, we need more information to reveal the scam hosting infrastructure. Since the spamvertised website usually short lived, we need to collect the information as we receive the spam email. As the results, we decided to build up our own spam email dataset.

In this chapter, we first introduce our spam email dataset including our novel data collection framework. Then we compare socware with spam email from three different perspectives, 1) supporting ecosystem, 2) intention of spam-advertised websites, and 3) incentives of spam email/posts.

5.1 Spam email dataset

Before we describe details of our dataset, we define the terminology we used first.

5.1.1 Terminology

Spammers employ unique infrastructures to distribute and profit from spam email. We divide any such spamming infrastructures into two distinct systems, "**Spam Distribution Infrastructure**" and "**Scam Hosting Infrastructure**." The spam distribution infrastructure includes all steps a spammer completes in order to deliver spam to a recipient, whereas the scam hosting infrastructure focuses on components necessary to enable the money making scams inherent to email spam.

Typical anti-spam technologies focus on the spam distribution infrastructure, preventing spam messages from ever reaching an end-user's inbox. For example, common anti-

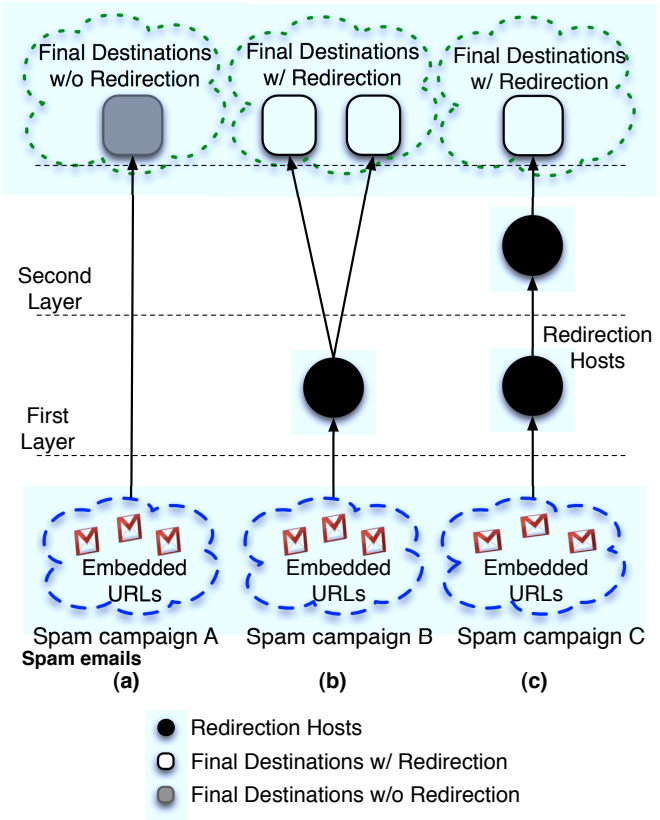


Figure 5.1: Relation between spam emails and final destination websites

spam technologies include pattern-based filtering, crowd-sourced filtering and blacklisting. Unfortunately, due to the overwhelming quantity and variation of spam email messages, disrupting the distribution infrastructure has done little to curb the global spamming business.

Recently, researchers have proposed alternative approaches to combat spam email, specifically designed to target the scam hosting infrastructure; that is, reducing the profitability of spam email [52] [113] [57] [73] [118]. In these works, the researchers identified a common theme of scam hosting infrastructures, which is, that in addition to providing an avenue for profiting on spam email, scam hosting infrastructure commonly decouple the

embedded URLs from the final destination websites by employing cooperative redirection hosts. Figure 5.1 depicts the possible scam hosting infrastructure spammers use to host their websites.

A spam campaign's goal is to direct end-users to a host that manages the spammer's scam. A scam may take many forms, including direct marketing campaigns, malicious software distribution, click-fraud and many others. We call URLs in the spam email bodies *embedded URLs*. We call the websites that end-users are ultimately directed to *Final Destinations*. Spammers decouple *embedded URLs* from final destinations by using cooperative *redirection hosts*. We further define a *redirection chain* as the set of redirection hosts. The redirection chain redirects the end-user through a series of redirection hosts, ultimately ending at the final destination. Now, depending on whether there are redirection hosts that decouple *final destinations* from *embedded URLs*, we may classify *final destinations* into two categories: (a) *Final Destinations without Redirection* and (b) *Final Destinations with Redirection*.

5.1.2 Spam feed

We use two month-long, publicly available spam data sets from Untroubled.org [36]. From the work of Anderson et al. [52], we know (a) that a spam campaign's lifetime is typically short and (b) usually final destination websites are taken offline shortly after a spam campaign is over. As a result, we use recently gathered spam messages as a spam feed and mine valid, online final destination websites, rather than websites that have expired.

5.1.3 Data Collection Framework

We build a data collection framework that takes as input spam messages, extracts the advertised URLs, follows each link to its final destination, probes all cooperating redirection hosts along the way. In particular, we use the Win32 API to directly control and monitor MS Internet Explorer [56]. Using an actual web browser distinguishes our framework from previous works, such as [52], by allowing us to follow both client- and server-side redirection methods, including JavaScript redirections.

In order to compare embedded websites against senders of spam email, we extract the spam's "sender" field from the header and collect the same information for each sender's IP address. However, it is easy for a spammer to falsify the email header information to avoid disclosure. As a result, we can only trust the IP address that established the SMTP connection with our mail server (i.e. the last relay host). In this study, we consider this IP address as the source IP address of the spam email.[1]

5.1.4 General Observations of spam email datasets

We analyze two month's worth of data, August and December 2009. In this chapter, we present results strictly for the December data set and we use August to support and contrast where necessary.

For each month, we extract roughly $47,000$ and $51,000$ unique embedded URLs, respectively. Because we collect information of each URLs in the next month after we collect the spam emails, some of URLs don't exist anymore. In our data sets, there are 47.37% and 64.07% of all unique embeddedURLs still can resolve to IP addresses ,respectively. These

[1]This method has already been adopted by previous works [104, 79, 120].

Table 5.1: general statistics of data set

	December 2009	August 2009
Number of Emails	49,881	72,524
Number of unique URLs	46,881	51,146
% of emails that contain URLs	70	76.23
% of unique URLs still on-line when probing	47.34	64.07
avg. redirection chain length	0.7	0.64
Number of final destination w/o redirection	4,138	3,934
Number of final destination w/ redirection	450	356
Number of redirection host	7,752	10,305

unique and live embedded URLs resolve to $11,964$ and $14,921$ unique websites. For each extracted URL, we trace each hop until the final destination, thus creating the redirection chain. Along the way, we probe each host in the redirection chain and query the WHOIS servers to get their basic informations. Table 5.1 shows the general statistics of our data.

We analyze the data in two ways: (a) from the perspective of the end-user (i.e., the beginning of the chain) and (b) from the perspective of the final destination (i.e., the end of the chain). When considering the beginning of the chain, we find that 64.75% of embedded URLs direct the end-user to a redirection host. However, when examining the end of the chain, we observe that only 11.33% of the final destination websites result from redirection chains. Most (88.67%) final destination websites are not obfuscated by redirection chains (see Fig. 5.2). This observation differs from the 2007 finding of Anderson et al.[52] that over 68% of scams used some kind of forwarding to obfuscate the final destination websites. One reason for this may be the different timeframes of observation (2007 versus 2009).

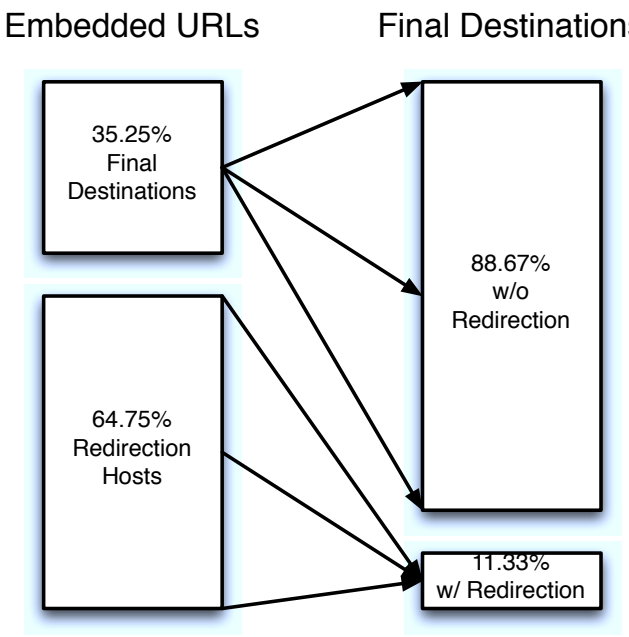


Figure 5.2: Aggregation of embedded URLs to final destinations. We see that redirection hosts are used to protect a small percentage of the overall final destinations.

5.2 Comparison between social malware and email spam

We compare socware with email spam from different perspectives. We mainly focus on three perspectives: 1) Supporting ecosystem, 2) Intention of spamvertised websites, and 3) Incentives of spam emails and socware posts. The comparisons are described in the following subsections.

5.2.1 Supporting ecosystem

There are two main parts of the supporting ecosystem for spam. One is "Spam Distribution Infrastructure" which includes all steps a spammer completes in order to deliver spam to a recipient. The other part is "Scam Hosting Infrastructure", which focuses on

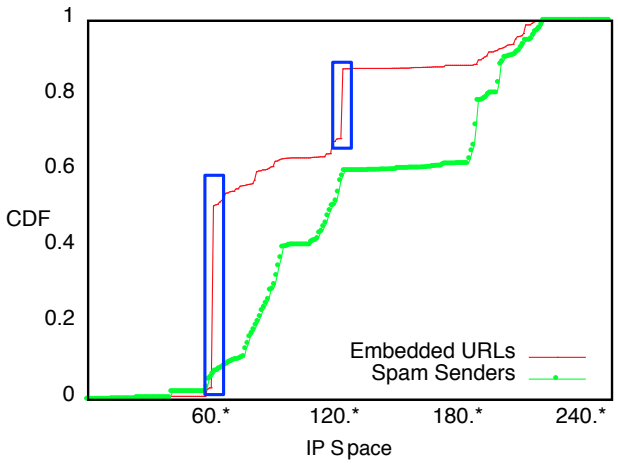


Figure 5.3: IP distributions of spam senders and embedded URLs (The boxes show highly active IP ranges of embedded URLs)

components necessary to enable the money making scam inherent from spam. In this section, we want to real the location of these infrastructures. More specifically, we want to answer these four questions: 1) Where does spam come from? 2) Where are spamvertised websites? 3) Where are the redirection hosts that separate the final destination websites from spam emails and socware posts? 4) Where are these final destination websites? We answer these questions by their network locations and geographical locations.

Spam email

Network location

We start with examining the distribution of embedded websites and spam email senders across IP space. Fig. 5.3 presents the CDF of IP addresses resolved from embedded URLs and spam email sender IPs. The IP addresses resolved from embedded URLs consists of the "first-hop", which may be either a final destination (in the case of final destinations

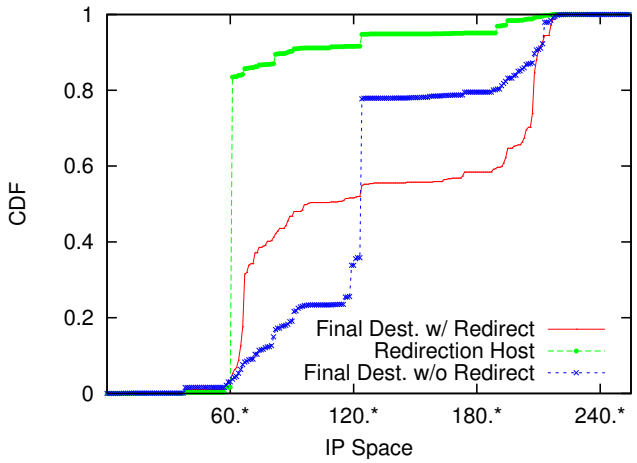


Figure 5.4: IP distribution of final destination sites and redirection hosts

without redirection) or the first redirection host (in the case of final destinations with redirection). From this figure, we observe that the IP addresses of embedded URLs are more highly concentrated across the IP space than spam email sender IPs. Specifically, spam email sender IPs are concentrated in four ranges of IP space, which constitute 29.41% of all spam email sender IPs and produce 88.2% of the total number of spam messages in our data set. In comparison, 70.39% of the embedded URLs are localized in two IP ranges (61.0.0.0/8 and 124.0.0.0/8).

As shown in Fig. 5.4, the distribution of final destination IPs without redirection and final destination IPs with redirection are quite different, with final destination IPs without redirection more concentrated in one IP range, whereas final destination IPs with redirection are more diffuse, yet exhibit two significant IP ranges. Further analysis of the data reveals two IP addresses *virtually hosting* [5] thousand of websites as final destinations without redirection and redirection hosts themselves. In final destinations without redirection, 49.42% of the domain names resolve to the same IP address (124.248.32.48). For redi-

Table 5.2: The Top Ten Countries of Spammers

Embedded URLs		Spam Senders	
Country	Percentage	Country	Percentage
China	68.54	Brazil	10.64
United Sates	10.00	Russian Federation	7.32
Korea	4.81	India	7.13
Russian Federation	3.09	China	5.86
Czech Republic	2.18	United Sates	5.80
United Kingdom	1.67	Vietnam	5.74
Germany	1.49	Korea	4.61
Ukraine	1.40	Argentina	2.89
Venezuela	1.04	Colombia	2.65
Turkey	0.78	Romania	2.56

rection hosts, 69.14% of the domain names resolve to the same IP address (61.235.117.75). Of all spam emails, 20.15% contain live URLs that advertise these two IP addresses.

Discussion. Modern anti-spam software commonly use the sender's IP address as one of the features to identify spam emails and in order to evade detection, spammers often employ broader ranges of the IP space. We suspect such evasion techniques result in the observed IP address distribution of email senders. Yet spammers seem less concerned with obfuscating their network locations of the scam hosting infrastructure, which is evident in the narrow IP ranges of embedded URLs and redirection hosts. This observation is supported by the conclusions of Kokkodis and Faloutsos [85].

Geographic location

In Table 5.2, we show the top ten countries in which spam senders and embedded URLs are located. Clearly evident is a strong concentration of embedded URLs in China,

Table 5.3: The percentage of servers in the same location

Kind of server	Country	AS
Sender and final destinations	9.57%	2.55%
Redirection hosts and final destination w/ redirection	79.21%	74.19%

and to a lesser degree, the United States. In contrast, the country origins of spam senders are more evenly distributed. Furthermore, the top 10 countries, which cover 95% of embedded URLs, only cover 32.20% of email senders. This observation supports our previous inference that spammers use distributed IPs to send spam emails in order to evade detection. By resolving embedded URLs to IP addresses, we find that most sites hosted in China are concentrated to a few IP addresses. Specifically, while China commands the lion's share of embedded URLs, only 3.08% of the IP addresses resolved from embedded URLs belong to China.

Furthermore, we find that the location of spam email senders differs from the location of embedded URLs in 90.43% of the email. Additionally, when redirection is considered, 79.21% of the embedded URLs are located in the same country as their redirection hosts (refer to Table 5.3).

Table 5.4 shows the distribution of countries of final destinations (with and without redirection) and redirection hosts. China is the most popular country to host both final destinations without redirection and redirection hosts, whereas the US is the most popular final destination when considering redirection.

Discussion. One possible reason for the geographic skew of embedded URLs is that those countries provide the necessary hosting infrastructure and lenient political/legal

Table 5.4: Top 10 countries of different web hosts

Final destination w/o redirection		Final destination w/ redirection		Redirection host	
Country	Percentage	Country	Percentage	Country	Percentage
China	44.06	United States	60.75	China	86.27
United States	16.54	Japan	7.17	United States	4.25
Korea	12.17	China	6.27	United Kingdom	2.23
Russian Federation	6.47	Russian Federation	6.09	Venezuela	1.79
Czech Republic	5.12	France	2.69	Turkey	1.01
Germany	2.93	Germany	2.15	Ukraine	0.93
Ukraine	2.26	United Kingdom	1.97	Russian Federation	0.86
Japan	1.00	Ukraine	1.79	Germany	0.57
France	1.00	Canada	1.61	Czech Republic	0.31
United Kingdom	0.93	Netherlands	1.43	Puerto Rico	0.21

leanings regarding spam. Without a stable scam hosting infrastructure, end-users may not be able to access scam-related redirection hosts or final destinations, thus precluding spammers from the opportunity to defraud or otherwise compromise end-users. A more lenient political and legal view on Internet spam and scams allows the scam hosting infrastructure to continue unabated and unhindered by local/international interference.

Socware

Network location

Unlike email spam, the senders of socware are victims' friends or the pages they like. This makes a very big difference between socware and email spam spam in the distributed infrastructure. Since the senders of socware are Facebook users or Facebook pages,

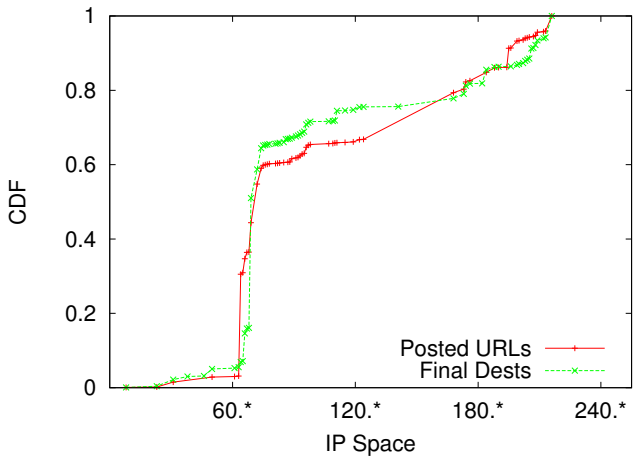


Figure 5.5: IP distribution of final destination sites and redirection hosts

it doesn't make sense to find their locations. They all will be in Facebook. Thus, we focus on the embedded URLs and the final destinations.

Figure 5.5 presents the CDF of IP addressees resolved from *Posted URLs* and *Landing URLs*. The IP addresses resolved from *Posted URLs* consist of the "first-hop", which may be final landing website, the first redirection hosts or the URL shortening service providers. From this figure, we observe that both of the Posted URLs and Landing URLs are highly concentrated in several IP spaces. Top 5 IP ranges (with prefix /8) cover around 63% of URLs. When we further check the ASes which host these URLs, we find that most of the landing URLs are hosted in Facebook. Moreover, most of the *Posted URLs* are hosted by several URL shortening service providers, for example, "tiny.url", "bit.ly", and etc. Table 3.9a and Table 5.5 list the top 10 ASes which host *Landing URLs* and *Resolved URLs*, respectively.

Geographic location

We list the top five countries which host *Posted URLs* and *Resolved URLs* in

AS Name	%
HURRICANE	23.98%
NTT America	12.77%
GOOGLE	11.2%
VeriSign	6.42%
SOFTLAYER	5.67%
DEEPMEDIA	5.12%
Hostway Services	3.35%
NETRIPLEX	2.6%
ThePlanet.com	2.32%
American Internet Services	2.32%

Table 5.5: Top 10 Posted URL ASes

Table 5.6 and Table 5.7. We observe the same trend that United States are the main country which host these URLs if we look at the top five countries that host *Landing URLs*, which is shown in Table 3.9c, together. One possible reason for the observation could be that most of the users in MyPageKeeper are in the United States. As the results, we might capture cascades, which specifically target people in the United States. In order to verify this conjecture, we need more socware that are collected from different countries. We leave this analysis as our future work.

Discussions

As we mention in the beginning of the section, there are two main parts of the supporting ecosystem for spam. One is "Spam Distribution Infrastructure" which includes all steps a spammer completes in order to deliver spam to a recipient. The other part is

Country	Percentage
United States	84.77%
Netherlands	5.94 %
Canada	1.64 %
United Kingdom	1.37 %
Thailand	1.30 %

Table 5.6: Top 5 countries hosting Posted URLs

Country	Percentage
United States	79.34%
Canada	2.82 %
Thailand	2.06 %
Lithuania	1.98 %
Russian Federation	1.45 %

Table 5.7: Top 5 countries hosting Resolved URLs

"Scam Hosting Infrastructure", which focuses on components necessary to enable the money making scam inherent from spam. We find that there are major differences between email spam and socware in these two infrastructures.

a) Scam Distribution Infrastructure: Spam emails are usually sent directly to users' inbox. On the contrary, socware usually cascades through online social network from users to users. The distribution method is one of the main differences between email spam and socware. When spammers can send spam email directly to different users' inboxes, socware developers rely on compromised users to spread socware

through the networks. As a result, socware developers need to put more effort on the content of the posts to make the most more attractive to users first. Then, socware can propagate from these users to their friends.

When majority of spam email are sent from botnets, majority of socware are sent from AppNets . As the distribution methods of email spam have advanced over the years and now, commonly, botnets are used to send over 87% of all spam [109]. We also observe that 44.7% of socware cascades are launched by Facebook applications (refer to 3.4.2). If we consider "plugin functions" also as a kind of Facebook applications, which is exactly implemented as applications by Facebook when the dissertation is written, these so-called "Facebook applications" are responsible for 78.1% of the socware cascades.

However, there is a major difference between botnets and AppNets. There are IP addresses associated with these bots. As a result, you can blacklist these bots or email servers which send spam emails. On the contrary, AppNets consist of compromised accounts in online social network. It is hard to blacklist these compromised accounts without the help of the OSN service providers. Some OSN service providers provide some functionalities so that users can create their own blacklists, but the effect is limited because of two reasons. First, only the users themselves can create their own blacklists. Second, they cannot share the blacklists with others friends. One of the main reasons that blacklists work well in protecting users from viruses or malicious websites is the collaboration between different blacklist providers or anti-virus companies.

b) Scam Hosting Infrastructure: Most of socware advertising websites are hosted in Facebook. We observe that 37% of URLs advertised by socware end up landing at Facebook domains 3.4.2, which make it harder to block socware by "blacklists".

As a result, socware developers don't need to set up redirection hosts to protect their websites which is popular in email spam [52]. On the contrary, socware developers usually just shorten the URLs of their websites so that the length of the post content doesn't exceed the character limits per post, which is set by the service providers [2].

The protection mechanisms for socware are highly restricted by the OSN service providers. We observe before that 44.7% of socware are sent by Facebook applications and 37% of the advertising websites are hosted on Facebook. This characteristic of socware makes third party Facebook application developers hard to protect Facebook users. The things they can do are highly restricted by the information provided by Facebook and the ability that Facebook authorize to their applications. For example, application developers might want to develop applications which can protect users from installing socware on Facebook. However, as a Facebook application, it can only flag or delete the socware posts on users timeline or newsfeed, which is what MyPageKeeper does. It cannot warn users at be beginning of installation process of socware, like what anti-virus software does when anti-virus software thought the installed program could be malicious.

5.2.2 Intentions

Spam Email

There are several studies [49, 50, 81] about the spamvertised websites. Here we are mainly focus on the categories of these websites. We list top five categories of spam emails reported by Symantec [49, 50] for the past three years in Table 5.8. We also list the portion of spam that these categories are responsible for in 2011 and 2010 for comparison.

[2]On Facebook, they have relax the characters limitation per post from 500 to 63206 [17].

Category	Percentage(2012)	Percentage (2011)	Percentage (2010)
Pharmaceutical	21.1%	39.6%	74%
Watches/Jewelry	9.2%	18.6%	6.5%
Sexual/Dating	54.6%	14.7%	3.3%
Unsolicited newsletters	7.4%	10.1%	9.3%
Jobs/Money Mules	4.4%	0.5%	-
phishing	0.4%	0.3%	-

Table 5.8: Top 5 categories of email spam [49, 50]

In Table 5.8, the percentage of "Pharmaceutical" keeps decreasing from 2010 to 2012. In 2010, it dominated the whole spam market. However, it only contributed to around 20% of the spam emails in 2012. On the other side, "Sexual/Dating" becomes the main category of email spam in 2012 although it only contributed to 3.3% of spam emails in 2010.

Socware

As we show in 3.4.3, "Phishing" and "Survey" are the main categories of websites advertised by socware. More specifically, 58.76% of socware advertising websites phish users' personal information. They either are Facebook applications which ask users to authorize them to access their Facebook profile, or they ask users to fill a form so that they can send the fake prize or gift to users. The other main portion of socware cascades (48.63%) benefit from asking users to do a survey. They lure users to finish surveys by fake rewards, like free iPad or gift cards.

Discussions

The intention of spamvertised websites by spam email and socware are

totally different. When most of the websites in spam emails sell users something or earn money from users by inviting users to a pay-per-view adult-content web cam sites, socware are mainly focus on phishing users personal information or earning money from asking users to finish surveys. As results shown in [81], users get what they purchase from these website advertised by spam emails. On the contrary, most of time users actually earn nothing instead of losing you credential and time when they follow the instructions on the website advertised by socware.

5.2.3 Incentives

After discussing about the intention of these spamvertised websites, we go back to the front side of the spam ecosystem, which is the incentives of spam emails or socware posts. This is an important factor for spammers business. If you cannot make users click the links in the spam emails or socware posts, users won't visit the spam-advertised websites. If they don't visit the websites, spammers earn nothing.

Spam email

Spam emails have become a very big business. "Pharmaceutical" and "Replica" and "Sexual/Dating" are the main categories of email spam. McCoy et al. [99] show that these three main categories are all associated with large complicated affiliate program networks, "partnerka," in Russia. It separates the advertisers from the sponsors in the affiliate program model. The sponsors are responsible for handling the design of websites, payment process and even customers services. As advertisers, they only need to focus on how to reach end users by either email delivery or search engine optimization. It also shows that email spammers don't need to figure out how to effectively attract users since

the products themselves and the competitive prices can attract users to click the links and purchase goods from these websites.

Socware

On the contrary, socware doesn't have a real product/service to attract users. Their main profit model is either asking users to finish surveys or luring users to install their application so that they can use these compromised accounts to promote others pages or websites. Rahman et al. [103] discuss an example model, called "like-as-a-service", where Facebook page or application owners pay socware developers for increasing the "number of like", or "number of install" on their pages/applications. In other words, socware do not provide any real product or service to users. As a result, socware developers have to use attractive incentive in the socware posts to entice users to click the links in the socware posts or install socware. From our study of incentives of socware posts, which are described in 4.3, most of the socware cascades use social incentives to entice users.

Among all social incentives sub categories, 44% of socware cascades exploit users' "social curiosity". An example of this category is a socware cascade with text content, "Check if a friend has deleted you". The socware exploits human's social curiosity by providing a fake Facebook functionality. These cascades successfully compromised 6578 users out of 15K MyPageKeeper users within 6 months.

The second largest categories in social incentives are *"Give a chance to support something or to show I care about something". An example socware cascade contains text content as " Do You Still Love The Steelers? Show Your Steelers Fever & Vote For Them As Your Favorite Team! "*. They abuse the enthusiasm of americans to National Football league (NFL) and successfully compromised 639 users in a very short time period

5.2.4 Summary

In this chapter, we show the different characteristics of socware and email spam. The differences are summarized here.

Support Ecosystem:

Spammers send spam emails directly to users' inboxes, like direct market. Socware cascades from users to users by word-of-mouth propagation method. Moreover, most of socware advertising websites are hosted in Facebook. It makes the traditional blacklist-based protection mechanism not work. Compared with email spammers who usually avoid their websites from exposing in the spam email directly by setting up redirection hosts in front of their websites, we observe socware developers protect their websites with less effort.

Intention of spam-advertised websites

We observe that the main categories of websites advertised by spam emails are "Pharmaceutical" "Watches/Jewelry" "Sexual/Dating". On the contrary, the main categories of socware cascades are "phishing" and "survey scam".

Incentives of spam email/posts

Email spammers usually join affiliate programs and earn money from the commission. As a result, they mainly focus on how to make spam email pass detection. As long as spam email can reach users' inboxes, the cheap goods, e,q., drugs, replicas, and computer software sold on the websites, attract users who need them to visit the spam-advertised websites automatically. On the contrary, socware developers exploit users "social curiosity" or "support to something they like" to spread their socware.

Chapter 6

Related Works

We discuss related efforts in brief.

Email spam campaigns. Different campaign definitions have been used in analyzing email spam campaigns. For example, Xie et al.[119] and Moore et al. [101] use the similarity of URLs embedded in emails, while others [102, 59] use email content similarity for grouping spam emails into campaigns. Similar to our definition of campaigns, Konte et al. [86] use landing URLs and Anderson et al. [52] use snapshots of landing webpages to identify email spam campaigns. In contrast to all of these efforts, our focus is on studying socware cascades, which display different characteristics from email spam [103, 72].

Analyzing spamming OSN accounts. Thomas et al. [110] characterized spamming accounts on Twitter and clustered spam campaigns based on the tweets sent from such accounts. Similarly, Benevenuto et al. [54] and Yang et al. [121] developed techniques to identify accounts of spammers on Twitter. Others have proposed a honey-pot based approach [108, 91] to detect spam accounts on OSNs. Yardi et al. [124] analyzed behavioral patterns among spam accounts in Twitter. Recently, Yang et al. [122] show that criminal

accounts on Twitter tend to be socially connected. Instead of focusing on accounts created by spammers, we analyze cascades that are predominantly propagated on Facebook by compromised users.

Detecting spam on OSNs. Gao et al. [72, 71] collect posts by crawling the walls of 3.5 million Facebook users and identify spam campaigns based on a combination of the text and the URLs appearing in posts. In a similar study on Twitter, Lee et al. [90] identified spam by grouping tweets into different topic groups based on the similarity of tweet contents. In our work, detection of spam is not of interest. We simply use a dataset of spam previously identified by MyPageKeeper, a Facebook security application, and use this dataset to systematically identify cascades and to analyze the characteristics of cascades.

Analyzing spam campaigns on OSNs. Gao et al. [72] analyzed spam campaigns on Facebook to identify the intention underlying these campaigns. However, they manually select keywords for each intention before grouping posts into campaigns. To understand spam campaigns on Twitter, Grier et al. [76] and Stringhini et al. [108] use landing URLs pointed to by spam tweets to identify campaigns. Unlike prior work, to the best of our knowledge, we present the first systematic analysis of how to identify cascades in the absence of landing URLs pointed to by posts on an OSN. We further analyze the provenance of spam posts and uncover the ecosystem of websites and applications underlying contemporary socware activities on today's most popular OSN—Facebook.

Measuring redirection characteristic and embedded URLs of spam emails Anderson et. al. [52] introduce *spamscatter*, a technique designed to identify a unique *scam* infrastructures. Their work focuses on correlating spamming campaigns to a specific scam hosting infrastructure, ignoring some redirection strategies, e.g. certain javascript redirec-

tion. In terms of this work, they study the relationship between final destination websites and spam email senders. Georgiou et. al. [73] examine the characteristics of HTML-based URLs in spam email. Again, their work ignores JavaScript URL redirection strategies. In contrast to these papers, our work focuses on redirection strategies employed in the scam hosting infrastructure by following all web browser redirection. Our work further studies the relationship between final destination websites and cooperative redirection hosts. Finally, Vangapandu et. al.'s [113] examine redirection strategies in general and contrasts legitimate redirections versus spam email redirection. They find that spam advertised websites use more JavaScript redirection and external redirection than legitimate websites.

Chapter 7

Conclusions

While Facebook, the largest online social networks, announces that they have over 1.11 billion users at March 2013, people in the United States spend 27% of their time on the Internet on online social networks [38]. Online social networks have started playing an important role in people's daily life. Spammers notice the trend and shift their attention from traditional email spam to this new playground, online social networks.

In the dissertation, we study socware through analysis of 173 billions posts collected from 16K users of MyPageKeeper for ten months. We uncover not only the ecosystem of socware, but also the incentive mechanisms used by socware to lure users into their traps.

Interestingly, we find that over half of socware cascades are propagated by users' clicking "Like/Share" on websites outside of Facebook, or by users' manually posting spam. It shows the carelessness of users in online social networks and the importance of user education to eliminate socware from Facebook but. Furthermore, we highlight the emergence of AppNet. It consist of Facebook applications which tightly collude with each other.

More interestingly, we show that socware targets human incentive mechanisms and

cognitive capacity to distinguish between a legitimate request from a friend and a bogus request from an infected friend. Moreover, we observe that socware acts like biological viruses, which use multiple incentives to get around users' diversity in susceptibility to distinct incentives. Our observations, supported with simulations, show that it an effective strategy to distribute socware in online social networks with multiple incentives. Last but not least, we show the very different characteristics of the two generations of online malicious behavior, socware and email spam.

Bibliography

[1] Ads to blame for malware in Facebook's Farm Town? http://news.cnet.com/8301-27080_3-20002267-245.html.

[2] Amazon Mechanical Turk. https://www.mturk.com/mturk/.

[3] Americans spend 23% of Internet time on social networks. http://mashable.com/2011/09/12/23-percent-online/.

[4] Anti-phishing working group. http://www.antiphishing.org/.

[5] Apnic. http://www.apnic.net/services/services-apnic-provides/helpdesk/faqs/virtual-web-hosting.

[6] Application authentication flow using oauth 2.0. http://developers.facebook.com/docs/authentication/.

[7] Documentation of facebook. http://developers.facebook.com/docs/reference/api/post/.

[8] Drive-by Download Attack on Facebook Used Malicious Ads. http://www.pcworld.com/businesscenter/article/241164/driveby_download_attack_on_facebook_used_malicious_ads.html.

[9] The economics of spam. http://www.aeaweb.org/articles.php?doi=10.1257/jep.26.3.87.

[10] Escrow-fraud. http://escrow-fraud.com/.

[11] F1 score. http://en.wikipedia.org/wiki/F1_score.

[12] Facebook birthday T-shirt scam steals secret mobile email addresses. http://nakedsecurity.sophos.com/2011/09/08/facebook-birthday-t-shirt-scam-steals-secret-mobile-email-addresses/.

[13] Facebook: Key facts. http://newsroom.fb.com/content/default.aspx?NewsAreaId=22.

[14] Facebook Opengraph API. http://developers.facebook.com/docs/reference/api/.

[15] Facebook socware incentive survey. http://goo.gl/PFCwe.

[16] Facebook stat 2011. http://www.kenburbary.com/2011/03/facebook-demographics-revisited-2011-statistics-2/.

[17] Facebook status limit jumps to 63,206 characters. http://goo.gl/cnjeYM.

[18] FBML- Facebook Markup Language. https://developers.facebook.com/docs/reference/fbml/.

[19] Firefox JSSH extension. http://croczilla.com/bits_and_pieces/jssh/.

[20] FireWatir. http://code.google.com/p/firewatir/.

[21] The global broadband speed test.

[22] Google Safe Browsing API. http://code.google.com/apis/safebrowsing/.

[23] Hackers selling $25 toolkit to create malicious Facebook apps. http://zd.net/g28HxI.

[24] How to fight socware - malware on facebook and other social networks. http://bit.ly/TJvMLd.

[25] How to spot a Facebook Survey Scam. http://facecrooks.com/Safety-Center/Scam-Watch/How-to-spot-a-Facebook-Survey-Scam.html.

[26] Joewein: Fighting spam and scams on the Internet. http://www.joewein.net/.

[27] Levenshtein distance. http://en.wikipedia.org/wiki/Levenshtein_distance.

[28] Malicious facebook app infects 5 million in 48 hours. http://www.itbusiness.ca/news/malicious-facebook-app-infects-5-million-in-48-hours/16121.

[29] MalwarePatrol- Malware is everywhere! . http://www.malware.com.br/.

[30] Most Common Surnames in the U.S. http://names.mongabay.com/most_common_surnames.htm.

[31] MyPageKeeper. https://apps.facebook.com/mypagekeeper/.

[32] People on facebook. http://www.facebook.com/press/info.php?statistics.

[33] Phishtank. http://www.phishtank.com/.

[34] Rihanna video scam. "http://www.virteacon.com/2011/11/sick-i-just-hate-rihanna-after-watching.html".

[35] Sample of conducteds survey. http://goo.gl/FUSNd.

[36] Spam archive. http://untroubled.org/spam/.

[37] Spamcop. http://www.spamcop.net/.

[38] Study: 27http://marketingland.com/study-27-of-time-online-in-the-us-is-spent-on-soc

[39] SURBL. http://www.surbl.org/.

[40] SVM Tutorials. http://svms.org/tutorials/.

[41] Team-Cymru. http://www.team-cymru.org/.

[42] URIBL. http://www.uribl.com/.

[43] Users of social networking websites face malware and phishing attacks. http://www.symantec.com/connect/blogs/users_social_networking_websites_face_malware_and_phishing_attacks.

[44] Web-of-trust. http://www.mywot.com/.

[45] What 20 Minutes On Facebook Looks Like. http://tcrn.ch/KytqzB.

[46] Your average Facebook post only reaches 12% of your friends. http://techcrunch.com/2012/02/29/facebook-post-reach-16-friends/.

[47] Zeus botnet targets facebook. http://blog.appriver.com/2009/10/zeus-botnet-targets-facebook.html.

[48] Spamtrackers. http://spamtrackers.eu/wiki/index.php/Yahoo_Groups, 2009.

[49] Internet security threat report, vol 17. http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_2011_21239364.en-us.pdf, 2012.

[50] Internet security threat report, vol 18. 'http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v18_2012_21291018.en-us.pdf', 2013.

[51] MW Adams, AH Ellingboe, and EC Rossman. Biological uniformity and disease epidemics. *BioScience*, pages 1067–1070, 1971.

[52] D.S. Anderson, C. Fleizach, S. Savage, and G.M. Voelker. Spamscatter: Characterizing internet scam hosting infrastructure. In *USENIX Security*, 2007.

[53] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*, pages 519–528. ACM, 2012.

[54] Fabricio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. Detecting spammers on Twitter. In *CEAS*, 2010.

[55] T Berners-Lee, L Masinter, and M McCahill. Rfc1738: Uniform resource locators (url), Jan 1994.

[56] Mayukh Bose. The world of mayukh bose. `http://www.mayukhbose.com/python/IEC/index.php`.

[57] T Bragin. Measurement study of the web through a spam lens. Technical Report TR-2007-02-01, University of Washington, Computer Science and Engineering., 2007.

[58] A.Z. Broder, S.C. Glassman, M.S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 1997.

[59] P.H. Calais, D.E.V. Pires, D.O. Guedes, W. Meira Jr, C. Hoepers, and K.S. Jessen. A campaign-based characterization of spamming strategies. In *CEAS*, 2008.

[60] Meeyoung Cha, Hamed Haddadi, Fabricio Benevenuto, and P Krishna Gummadi. Measuring user influence in twitter: The million follower fallacy. *ICWSM*, 10:10–17, 2010.

[61] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2, 2011.

[62] Kumar Chellapilla and Alexey Maykov. A taxonomy of javascript redirection spam. In *AIRWeb '07: Proceedings of the 3rd international workshop on Adversarial information retrieval on the web*, pages 81–88, New York, NY, USA, 2007. ACM.

[63] Duanbing Chen, Linyuan Lü, Ming-Sheng Shang, Yi-Cheng Zhang, and Tao Zhou. Identifying influential nodes in complex networks. *Physica A: Statistical Mechanics and its Applications*, 391(4):1777–1787, 2012.

[64] Jun-Jun Cheng, Yun Liu, Bo Shen, and Wei-Guo Yuan. An epidemic model of rumor diffusion in online social networks. *The European Physical Journal B*, 86(1):1–7, 2013.

[65] Prabhakar Raghavan Christopher D. Manning and Hinrich Schütze. Introduction to information retrieval. `http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html`.

[66] Graham Clule. Justin bieber stabbed by a crazed fan? it's a facebook scam. `http://nakedsecurity.sophos.com/2011/12/08/justin-bieber-stabbed-facebook-scam/`.

[67] Fred Cohen. Computer viruses: theory and experiments. *Computers & security*, 6(1):22–35, 1987.

[68] Nicole B Ellison, Charles Steinfield, and Cliff Lampe. The benefits of facebook friends: social capital and college students use of online social network sites. *Journal of Computer-Mediated Communication*, 12(4):1143–1168, 2007.

[69] R Ford and E H Spafford. Computer science: Happy birthday, dear viruses. *Science*, 317(5835):210–211, July 2007.

[70] Stephanie Forrest, Steven A Hofmeyr, and Anil Somayaji. Computer immunology. *Communications of the ACM*, 40(10):88–96, 1997.

[71] Hongyu Gao, Yan Chen, Kathy Lee, Diana Palsetia, and Alok Choudhary. Towards online spam filtering in social networks. In *NDSS*, 2012.

[72] Hongyu Gao, Jun Hu, Christo Wilson, Zhichun Li, Yan Chen, and Ben Y Zhao. Detecting and characterizing social spam campaigns. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 35–47. ACM, 2010.

[73] E Georgiou, M Dikaiakos, and A Stassopoulou. On the properties of spam-advertised url addresses. *J. Netw. Comput. Appl.*, 31(4):966–958, 2008.

[74] Michaela Goetz, Jure Leskovec, Mary McGlohon, and Christos Faloutsos. Modeling blog dynamics. In *ICWSM*, 2009.

[75] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: The underground on 140 characters or less. In *CCS*, 2010.

[76] Chris Grier, Kurt Thomas, Vern Paxson, and Michael Zhang. @spam: The underground on 140 characters or less. In *CCS*, 2010.

[77] Ting-Kai Huang, Md Sazzadur Rahman, Harsha Madhyastha, Michalis Faloutsos, and Bruno Ribeiro. An analysis of socware cascades in online social networks. In *WWW*, 2013.

[78] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 1985.

[79] John P. John, Alexander Moshchuk, Steven D. Gribble, and Arvind Krishnamurthy. Studying spamming botnets using botlab. In *NSDI 2009: Proceedings of the 6th Symposium on Networked Systems Design & Implementation*, Boston, MA, April 2009.

[80] Daniel Kahneman. *Thinking, fast and slow.* Farrar, Straus and Giroux, 2011.

[81] 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*, pages 3–14. ACM, 2008.

[82] Gregg Keizer. Worm spreads on facebook, hijacks users' clicks. `http://www.computerworld.com/s/article/9122724/Worm_spreads_on_Facebook_hijacks_users_clicks`.

[83] Maksim Kitsak, Lazaros K Gallos, Shlomo Havlin, Fredrik Liljeros, Lev Muchnik, H Eugene Stanley, and Hernán A Makse. Identification of influential spreaders in complex networks. *Nature Physics*, 6(11):888–893, 2010.

[84] Bryan Klimt and Yiming Yang. Introducing the enron corpus. In *First conference on email and anti-spam (CEAS)*, 2004.

[85] M Kokkodis and M Faloutsos. Spamming botnets: Are we losing the war? In *CEAS 09: Proceedings of 6th Conference on Email and Anti-Spam*, Mountain View, California, July 2009.

[86] M. Konte, N. Feamster, and J. Jung. Dynamics of online scam hosting infrastructure. In *PAM*, 2009.

[87] M Konte, N Feamster, G Tech, and J Jung. Fast flux service networks: Dynamics and roles in hosting online scams. *cc.gatech.edu*.

[88] 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 *LEET*, 2009.

[89] Anh Le, Athina Markopoulou, and Michalis Faloutsos. Phishdef: Url names say it all. In *Infocom*, 2010.

[90] K. Lee, J. Caverlee, Z. Cheng, and D.Z. Sui. Content-driven detection of campaigns in social media. In *CIKM*, 2011.

[91] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: social honeypots + machine learning. In *SIGIR*, 2010.

[92] Sangho Lee and Jong Kim. Warningbird: Detecting suspicious urls in twitter stream. *NDSS*, 2012.

[93] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506. ACM, 2009.

[94] V Levenstein. Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, 1(1):8–17, 1965.

[95] M. Li, X. Chen, X. Li, B. Ma, and P.M.B. Vitányi. The similarity metric. *Information Theory, IEEE Transactions on*, 2004.

[96] David Liben-Nowell and Jon Kleinberg. Tracing information flow on a global scale using internet chain-letter data. *Proceedings of the National Academy of Sciences*, 105(12):4633–4638, 2008.

[97] Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. Beyond blacklists: learning to detect malicious web sites from suspicious urls. In *KDD*, 2009.

[98] Winter Mason and Siddharth Suri. Conducting behavioral research on amazon's mechanical turk. *Behavior research methods*, 44(1):1–23, 2012.

[99] Damon McCoy, Andreas Pitsillidis, Grant Jordan, Nicholas Weaver, Christian Kreibich, Brian Krebs, Geoffrey M Voelker, Stefan Savage, and Kirill Levchenko. Pharmaleaks: Understanding the business of online pharmaceutical affiliate programs. In *Proceedings of the 21st USENIX conference on Security symposium*, pages 1–1. USENIX Association, 2012.

[100] G.W. Milligan and M.C. Cooper. A study of the comparability of external criteria for hierarchical cluster analysis. *Multivariate Behavioral Research*, 1986.

[101] T. Moore, R. Clayton, and H. Stern. Temporal correlations between spam and phishing websites. In *LEET*, 2009.

[102] F. Qian, A. Pathak, Y.C. Hu, Z.M. Mao, and Y. Xie. A case for unsupervised-learning-based spam filtering. In *ACM SIGMETRICS Performance Evaluation Review*, 2010.

[103] Md Sazzadur Rahman, Ting-Kai Huang, Harsha V Madhyastha, and Michalis Faloutsos. Efficient and scalable socware detection in online social networks. In *USENIX Security*, 2012.

[104] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. *SIGCOMM Comput. Commun. Rev.*, 36(4):291–302, 2006.

[105] Eugene H Spafford. Computer viruses as artificial life. *Artificial Life*, 1(3):249–265, 1994.

[106] Spamlaws. Spam statistics and facts. `http://www.spamlaws.com/spam-stats.html`.

[107] T. Stein, E. Chen, and K. Mangla. Facebook immune system. In *SNS*, 2011.

[108] G. Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *ACSAC*, 2010.

[109] Symantec. State of spam, a monthly report, Nov 2009.

[110] K. Thomas, C. Grier, D. Song, and V. Paxson. Suspended accounts in retrospect: an analysis of twitter spam. In *IMC*, 2011.

[111] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. Design and Evaluation of a Real-Time URL Spam Filtering Service. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2011.

[112] AJ Ullstrup. The impacts of the southern corn leaf blight epidemics of 1970-1971. *Annual Review of Phytopathology*, 10(1):37–50, 1972.

[113] K Vangapandu, Douglas Brewer, and Kang Li. A study of url redirection indicating spam. In *CEAS 09: Proceedings of 6th Conference on Email and Anti-Spam*, Mountain View, California, 2009.

[114] John Von Neumann and Arthur Walter Burks. *Theory of self-reproducing automata*. University of Illinois press Urbana, 1966.

[115] P Wang, M C Gonzalez, C A Hidalgo, and A L Barabasi. Understanding the Spreading Patterns of Mobile Phone Viruses. *science*, 324(5930):1071–1076, May 2009.

[116] YM Wang, M Ma, Y Niu, and H Chen. Spam double-funnel: Connecting web spammers with advertisers. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 291–300, New York, NY, USA, 2007. ACM.

[117] C Wei, A Sprague, G Warner, and A Skjellum. Characterization of spam advertised website hosting strategy. In *CEAS 09: Proceedings of 6th Conference on Email and Anti-Spam*, Mountain View, California, July 2009.

[118] B. Wu and B.D. Davison. Cloaking and redirection: A preliminary study. In *First International Workshop on Adversarial Information Retreival on the Web (AIRWeb)*, 2005.

[119] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: signatures and characteristics. In *CCR*, 2008.

[120] Yinglian Xie, Fang Yu, Kannan Achan, Eliot Gillum, Moises Goldszmidt, and Ted Wobber. How dynamic are ip addresses? In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 301—312, New York, NY, USA, 2007. ACM.

[121] Chao Yang, Robert Harkreader, and Guofei Gu. Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers. In *RAID*, 2011.

[122] Chao Yang, Robert Harkreader, Jialong Zhang, Seungwon Shin, and Guofei Gu. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter. In *WWW*, 2012.

[123] Jaewon Yang and Jure Leskovec. Modeling information diffusion in implicit networks. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 599–608. IEEE, 2010.

[124] S. Yardi, D. Romero, G. Schoenebeck, et al. Detecting spam in a twitter network. *First Monday*, 2009.

[125] Hsiang-Fu Yu, Chia-Hua Ho, Prakash Arunachalam, Manas Somaiya, and Chih-Jen Lin. Product title classification versus text classification. Technical report, Technical report, 2012. URL http://www. csie. ntu. edu. tw/˜ cjlin/papers/title. pdf.

[126] Hsiang-Fu Yu, Chia-Hua Ho, Yu-Chin Juan, and Chih-Jen Lin. Libshorttext: A library for short-text classification and analysis.