UNIVERSITY OF CALIFORNIA, SAN DIEGO

Verilogo: Proactive Phishing Detection via Logo Recognition

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Computer Science

by

Ge Wang

Committee in charge:

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

2010

The thesis of Ge Wang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____
Co-Chair

_____
Co-Chair

University of California, San Diego

2010

DEDICATION

To my parents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Chapter 1, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 2, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 3, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 4, in full, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 5, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 6, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 7, in full, has been submitted for publication of the material as it may ap-

pear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

ABSTRACT OF THE THESIS

Verilogo: Proactive Phishing Detection via Logo Recognition

by

Ge Wang

Master of Science in Computer Science

University of California, San Diego, 2010

Professor Hovav Shacham, Co-Chair
Professor Stefan Savage, Co-Chair

Defending users against fraudulent Web sites (i.e., phishing) is a task that is reactive in practice. Blacklists, spam filters and takedowns all depend on first finding new sites and verifying that they are fraudulent. In this thesis we explore an alternative approach that uses a combination of computer-vision techniques to proactively identify likely phishing pages as they are rendered, interactive queries to validate such pages with brand holders, and a single keyboard-entry filter to minimize false positives. We have developed a prototype version of this approach within the Firefox browser and evaluate it for both accuracy and performance. While no such approach is perfect, we believe our technique offers a significant new capability for minimizing response time in combating a wide range of phishing scams.

Chapter 1

Introduction

With the growth in popularity of web-based retail, online banking and social networking, the security of one's online identity has become more critical than ever. In recent years, internet fraud has been on the rise, and attackers have made profits of more than $250 million in 2008 alone [1]. The relative ease with which an attacker can create and distribute phishing content on the web, in conjunction with the difficulty of finding, verifying and taking down malicious content, have given criminals an edge in successfully defrauding Internet users. Anti-phishing efforts must constantly evolve to defend against ever more sophisticated phishing attacks.

In this thesis, we introduce on-line vision-based phishing detection as a technique for combating web-based fraud. By using a combination of computer vision techniques, brand holder validation and phishing detection heuristics, we provide an approach which will help decrease the turnaround time for blacklist population and add a client-side line of defense against currently available phishing attacks. To optimize the performance and versatility of the phishing detection, we focus on performing image matching on brand holder logos, which are frequently used as an essential trust cue in phishing efforts. Using the results from the image matching, we apply additional techniques to verify the authenticity and credibility of a web page.

The attraction of this approach is that it can be performed proactively on each user's computer and thus can offer protection against "zero-hour" phishing sites that have not yet been discovered or categorized. We have realized this idea within an extension to the Firefox browser called Verilogo. It uses computer-vision algorithms to

automatically identify the presence of heavily-phished logos as each Web page is rendered, implements a protocol for online validation of such pages with the associated brand holders, and uses a simple keyboard-entry heuristic to filter out sites making non-fraudulent, but unauthorized use of a brand logo. This thesis describes our design and implementation of our phishing detection mechanisms, as well as our experiences with our prototype implementation.

In the remainder of this thesis, we describe this technique and our experience with our prototype Verilogo implementation. In Chapter 2 we provide more background on fraudulent sites and describe related work focused on combating it. Next, in Chapter 3 we provide background on the SIFT algorithm, our adaptation for use in logo matching, and an evaluation in a controlled setting — showing both logo separation and the ability to match transformed images from a single reference logo. In Chapter 4 we describe the Verilogo prototype and its performance characteristics, followed by an evaluation of this prototype on a variety of real pages (brand owned sites, phishing sites, popular sites and random sites) in Chapter 5. Finally, we discuss opportunities and practical challenges in using this approach in Chapter 6 and then summarize our overall findings.

Chapter 1, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 2

Background

Over the last few decades, internet fraud has become an increasingly lucrative business for online attackers. Ease of information distribution and the popularity of e-commerce and online banking have provided a fertile environment for scammers to profit from unsuspecting users. In this section, we provide a brief background on phishing, one of the most popular techniques used to commit internet fraud. We discuss how phishing is perpetrated and current techniques to defend against it.

2.1 Fraud

While the idea of fraud is hardly a new one, the growth of internet technology in the last few decades have provided defrauders with unprecedented opportunities. Almost every internet user has, at some point, been targeted (successfully or not) by unscrupulous people trying to obtain credentials, credit card information, and bank account numbers. For some victims, the result of the fraud may simply be inconvenience (i.e. being locked out of an email account), while for others, the consequences may be far more sever. In 2008, the Internet Crime Complaint Center (IC3) received over a quarter of a million complaints from defrauded victims, who lost a total of over $250 million [1]. Aside from monetary losses, defrauded victims may also suffer from the many lasting repercussions of identity theft and loss of personal or sensitive information. For example, a popular scam in 2009 saw perpetrators using stolen email and social networking accounts to send messages to the victims' friends, claiming emergency or distress and

asking for money. Beyond losing control of an email account or financial loss, victims may also see emotional distress and damage to their reputation or credibility.

Although fraud comes in many flavours, among the most popular is phishing fraud. Phishing is an attempt to illicitly gather sensitive information (account logins, credit card numbers, bank account information etc.) about victims, usually by posing as a legitimate authority. Phishers often establish a website ostensibly belonging to a brand holder (such as Bank of America), and prompt users to enter their account information. Web sites are specifically designed to emulate, if not duplicate, the look of the legitimate brand holder in an effort to lure potential victims. Worse yet, Internet users often give little scrutiny to potentially dangerous web pages. A study by Dhamija et. al. in 2006 indicated that 23% of users paid no attention to security indicators or the address and status bars, and that well-crafted phishing sites can fool 90% of users [9].

The cheapness and ease with which a fraudulent website can be set up has led to a disturbing pervasiveness of phishing attacks. No sooner has one phishing site been identified and taken down, more have taken its place. Takedown efforts are necessarily reactive, and the time between a website being put up and its being identified as phish, verified and taken down is more than enough for users to fall victim to the phishing attack. Thus, the lag time before a website is blacklisted is critical. Unfortunately, few blacklists have better than 20% coverage of new phishing sites found in spam emails, and it could take hours before more than 50% of the sites are identified [29].

2.2 Phishing Attacks and Defenses

To perpetrate an attack, a phisher will usually host a website that looks very similar to a brand holder's site, on a domain under her control. Many phishing sites will try to obtain domain names that resemble the name of the target brand holder site. One such technique, typo squatting, capitalizes on user typos. Thus, a phisher may attempt to purchase "facebok.com" in order to obtain traffic from users who have mistyped "facebook.com". The phishing site may be "advertised" in a number of ways, such as email spam or the typo squatting technique previously described. Once an unsuspecting user has been drawn to the fraudulent website, a phisher can simply wait and collect any

submitted credentials.

Since phishing pages generally must closely resemble a brand holder page to appear convincing, the simplest thing for a phisher to do is to copy the source code of the legitimate page and host it on his own domain. However, such pages can easily be identified as phish, since simply searching for websites with source code very similar to that of a legitimate page will uncover these pages. A more sophisticated attacker, however, can modify the page sufficiently to both fool potential victims and subvert source code matching.

Significant research has been done to identify and prevent increasingly clever phish. In general, offline detection mechanisms are simpler to implement and are important to take-down efforts, as they are not constrained by the need to identify phish in "real-time" as users are browsing the web. On-line detection mechanisms are much more sensitive to performance requirements, but they offer users defense against phishing attacks that have not yet been identified by blacklists. Browser-side defense mechanisms such as SpoofGuard [8] utilize a number of heuristics to calculate a score that determines the likelihood that a webpage is phish. Some heuristics and techniques that are used to detect phish include:

checking the referring vector many phishers lure users to their website using email spam, so websites that are accessed via an email link are considered more likely to be phish.

classifying URLs domain names and URLs that are very similar to the names of brand holders may represent typo-squatting, and thus may be considered suspicious.

comparing website "signatures" "signatures" are generated for popular websites and compared with potential phishing sites.

Although many current techniques have had great success in finding phish, inevitably each technique will yield a number of false positives (harmless pages incorrectly classified as phish) or false negatives (phishing pages that were not detected). The former case may be dismissed as a necessary nuisance, provided the number of false positives

is very small. However, even a small number of false negatives can do a lot of damage. Many anti-phish mechanisms don't catch every kind of existing phish. Furthermore, phishers are constantly introducing new techniques to subvert current defenses against phish.

There have been some efforts in the area of vision-based phishing detection. One example is work done by Kuan-Ta Chen et. al. [7], using the CCH image matching algorithm to compare the login pages of legitimate sites against suspected phishing sites. However, comparing the overall look and feel of the real and fake pages is not necessarily a good way to detect phish, as a webpage really does not need to look very similar to the original legitimate page to successfully fool users into giving up credentials. Our work differs in several respects, first by using the SIFT algorithm, which is more tolerant to slight aberrations and differences in the two images being compared, and secondly in that we focus exclusively on the logos used on a page rather than the look of the entire page as a whole. Envisional's ImageFlare is a commercial product that, similar to our work, performs image matching on brand holder logos. However, the intent is as a method for particular brand holders to protect the integrity of their brand by scouring the internet for usage of their logo. Such an application does not appear to be well-suited to performing phishing detection for popular brands without a prohibitive amount of computing resources. Furthermore, it would probably not provide any client-side online protection against phish as their techniques seem to favour offline analysis. In general, a significant downside to vision-based phishing detection is performance. Vision algorithms are often compute-intensive and thus difficult to integrate effectively into client-side protection mechanisms that require real-time performance guarantees.

Also closely related to our work is Verisign's Secure Internet Letterhead proposal [14], which would embed logos in X.509 certificates and present these logos to the user. The two approaches are complementary: Internet Letterhead allows users to gain confidence in authorized logos; our proposal identifies and alerts the user to unathorized logos.

Once phish has been detected, measures may be taken to prevent users from becoming victimized. Although most popular browsers contain a number of security features to deter phish, a number of studies have shown that most users ignore warn-

ings such as an unfamiliar domain name (even if the domain name is highlighted by the browser), certificate warnings and missing lock icons. Heuristic-based, client-side defense mechanisms provide a speedier turnaround time than blacklists, but may be limited in scope with respect to the types of phish they detect. New types of phishing attacks may quickly make the heuristics obsolete and increase the number of false positives or false negatives. As a result, users may become further desensitized to security alerts and simply "click through". A more large-scale method of preventing phish that requires no end user decision-making is to blacklist and take down phishing webpages, thus stopping the attack from reaching any users. However, many of these efforts require manual vetting of pages identified as potential phish. Waiting for a site to be submitted for consideration as phish and performing the check are potentially slow and costly operations at a large scale. A study by Moore and Clayton showed that on average, it may take 64 to 96 hours to take down a malicious website [25].

While these anti-phishing techniques represent a considerable body of work, we believe that ours provides a distinct contribution. Succinctly, we are focused on providing client-side detection of new phishing sites — thus offering the latency benefits of heuristics — but using a feature set (visual logo matching) that is both robust and produces far fewer false positives in practice. Although vision techniques have previously been applied to phishing detection, we believe that ours is the first that can be practically used to detect phish within a browser context without significant slowdown or degradation of a user's browsing experience.

Chapter 2, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 3

Logo-based phishing detection

Our approach consists of three steps: first, we identify whether a Web page contains a given logo. Second, we determine if the brand holder has authorized the hosting IP address to use its logo. Finally, for sites that both contain logos and are unauthorized we issue a warning only if the user enters keyboard input into the page. The image matching algorithm in the first step uses the SIFT algorithm developed by David Lowe [19] and is based on the Master's thesis work of Sebastian Becerra [5]. We first introduce our threat model, then describe each of these steps in turn.

3.1 Threat model and assumptions

There are many types of online fraud and a variety of attacker goals and capabilities. We will briefly describe the subset of these that our work will focus on. We assume that an attacker controls a domain on which to host his phishing site, and that the domain has not (yet) been blacklisted. Furthermore, the attacker has the ability to direct potential victims to his phishing site via one or more vectors (such as email spam or typo-squatting). We assume the goal of the attacker is to obtain sensitive information from a user by tricking them into entering their credentials into the attacker's malicious site. Other attacker goals and capabilities (such as attempting to install malware on a user's computer or exploiting an already-compromised computer to disable security capabilities) are considered to be out of scope for our work.

3.2 Image matching

To identify logos found on a web page, we must first create a database of brand holder logos, then compare the web page against the database to find potential matches. The simplest comparison method is simply performing a pixel-by-pixel comparison of the web page and each candidate logo. However, exact matching can be easily subverted by an attacker, who can simply modify the logo enough such that a user will be unable to tell the difference, but the matching algorithm fails. To obtain more robust matching, we use the Scale Invariant Feature Transform (SIFT) algorithm pioneering by David Lowe [19] to perform image comparisons. SIFT is tolerant of many image transformations, including rotation, shear, noise, scaling and color changes, making it much more difficult for an attacker to both "trick" the algorithm and still maintain a recognizable logo. To perform matching, SIFT first generates a set of features for each image (in our case, the logos and the web page). Then, the features of the image to be matched (the web page) are compared against the features of the images in the database (the logos). We will describe these steps in further detail.

3.2.1 Feature generation

There are four major computational steps required to extract features for an image:

Scale-space extrema detection This step searches for features that remain constant over different scales of the image. These scale-space extrema are obtained by first convolving the original image repeatedly with a Gaussian filter to produce a stack of images separated by a constant factor in scale space. Next, the Difference of Gaussians function is applied to each image in the stack, and extrema are determined by comparing each point with its neighbours in its own scale and adjacent scales.

Keypoint localization Unstable points are further pruned by fitting each point to a model for location and scale and rejecting points with low contrast or that are edge responses.

Orientation assignment An orientation histogram of 36 bins is then generated for each location using the gradient orientations from the surrounding area. One or more orientations corresponding to the largest peaks in the histogram are assigned to the location. Duplicate keypoints are created in cases where multiple orientations are used.

Keypoint descriptor A distinct descriptor is assigned to each keypoint by calculating image gradients with respect to the previously determined orientation assignment. The gradients are obtained from a 16 X 16 area around the keypoint and placed in an array of 4 X 4 histograms with 8 bins each (where each histogram covers a 4 X 4 subregion). Samples are smoothed by using trilinear interpolation, and the bin values from all 16 histograms are concatenated together to form the 128-element feature vector. This vector is then normalized to reduce the effects of affine changes in illumination.

The distinctive, scale-invariant properties of the features generated by SIFT are particularly suitable for use in logo matching. Since matching can be performed with a relatively small number of features, we are still able to accurately match many simplistic logos that do not yield many stable keypoints.

In our phishing detection algorithm, features for each of the brand holder logos are generated using this method and stored in a database. To find logos in a web page, we also generate features for the web page before performing the feature matching step, which we will describe in the following section.

3.2.2 Feature matching

Once features are generated for both the set of logos in the database as well as the web page, we compare the features of each logo against the features of the web page to obtain likely matches. This is done by calculating the Euclidean distances of the two closest neighbours of each keypoint. A web page's keypoint matches a logo's keypoint if the ratio between the two distances is below 0.8. To quickly determine the nearest two neighbours of a given keypoint, the Best-Bin-First heuristic due to Bies and Lowe [6] is used. In order to optimize performance for large logo databases, we use a coarse-to-fine approach similar to [26] to whittle the database down to a smaller number "likely"

candidate logos, then perform detailed matching on these.

In the end, this process produces a score for each logo (representing the keypoint match percentage) that we turn into a binary feature based on a threshold. As this threshold is increased, false positives decrease while a lower threshold permits "fuzzier" matching and hence fewer false negatives. In our implementation, we consider two regimes for threshold setting: one in which the threshold is set to a single value for all logos, which is simple but can generate unnecessary false positives, and one in which poorly performing logos (i.e., logos that are insufficiently unique) are given distinct thresholds.

3.3 Brand authorization

While the SIFT algorithm provides the means for robust image matching, this is not, by itself, an anti-phishing defense. As we alluded to earlier, the mere presence of a bank's logo on a page is insufficient to conclude that the site is fraudulent. Indeed, the page may be owned by the bank itself or may simply be a third-party site (e.g., a news site) that is displaying the logo incidentally and with no intent to defraud. We address these two cases independently.

To identify pages hosted by the brand holder (or its designates) we propose to *ask* them. We consider two potential approaches, inspired by the SPF [31] and DomainKeys [18] mechanisms used for similar purposes in binding IP addresses and the domains used in e-mail "From:" addresses. In both cases we assume that the browser is distributed with a logo database that may be further annotated with information such as the brand holder's registered domain and possibly a certificate. Given the relatively small number of brand holders who are actively phished, this assumption seems reasonable.

In the first case we define a new DNS record type, BAP (Brand Authorization Policy), that defines the list of IP addresses or domains authorized to display the brand holder's marks (using a syntax similar to SPF). Upon detecting that a particular logo is present on a page, the browser queries the registered domain for the associated brand holder and requests the BAP record. If the Web page is being hosted by a site address

contained in the BAP record then it is considered benign and all further processing stops. To bootstrap use, in the event that the BAP record is not found, the browser may default to use a local database of whitelist rules per brand.

The disadvantage of this approach is that it requires a DNS server update for each new site that the brand holder desires to become authorized. An alternative approach, modeled on DomainKeys, is to have brand holders embed a digital signature in their logos (e.g., in the GIF or PNG comment field) that covers both the image content and the hosting IP address. Upon finding a logo on a Web page, the browser then determines which image object was rendered at that location and validates it using the associated brand holder's public key (such keys can either be part of the browser's brand database or distributed via a separate PKI such as DNSSEC).

In our current prototype, we have built a simplified version of the BAP approach, but we believe implementing either technique is feasible; the logistical issues are similar to those involved in SPF and DomainKeys, which have seen wide adoption [10].

3.4 Input filtering

Unfortunately, popular logos are routinely used in news sites that report on the associated brands. Such sites are unlikely to be "authorized", and even if the brand holder wanted to authorize them, the challenge of tracking all sites making incidental use of a logo is likely insurmountable. Instead, we rely on a simple heuristic to filter out sites that might be "true phishing sites" from those that could not be. In particular, we track when a user provides keyboard input to an unauthorized page and use this event to trigger a warning.

We've chosen to explore this very simple heuristic for two reasons. First, we believe it is highly robust to evasion. Phishing sites need input to acquire user credentials and thus sites that are simply "viewed" do not place a user at risk. We have found few confirmed counter-examples to this rule, excepting a small number of phishing sites that request the user to download and run executable software (presumably a malware of some sort). In principal, one could easily add a "user attempted to download executable" heuristic to address these cases, but we do not evaluate that here.

Second, we've purposely selected a minimal part of the design space to make analysis straightforward and conservative. Its clear that false positives could be reduced even further using more advanced site heuristics or learning algorithms [20, 23, 4, 33, 35, 8, 13, 21, 17] but teasing apart their relative contribution and interaction would obscure the basic result we're trying to demonstrate. Moreover, each new heuristic can offer additional opportunities for evading detection that we are not prepared to evaluate here. For example, the impact of domain reputation systems has led phishers to host their pages on compromised sites (effectively bypassing any "low reputation hosting" heuristic). That said, we believe a widely-fielded system would inevitably combine our approach with other such measures to still further reduce its false positive rate.

3.5 Mashups

One type of site that poses special problem is a "mashup," such as iGoogle, that combines content from multiple legitimate sites. Brandholders may or may no be willing to participate in mashups. Those that are willing could authorize specific mashups by means of policy files analogous to Flash's `crossdomain.xml`; see, e.g., [16, 28].

Chapter 3, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 4

Verilogo browser

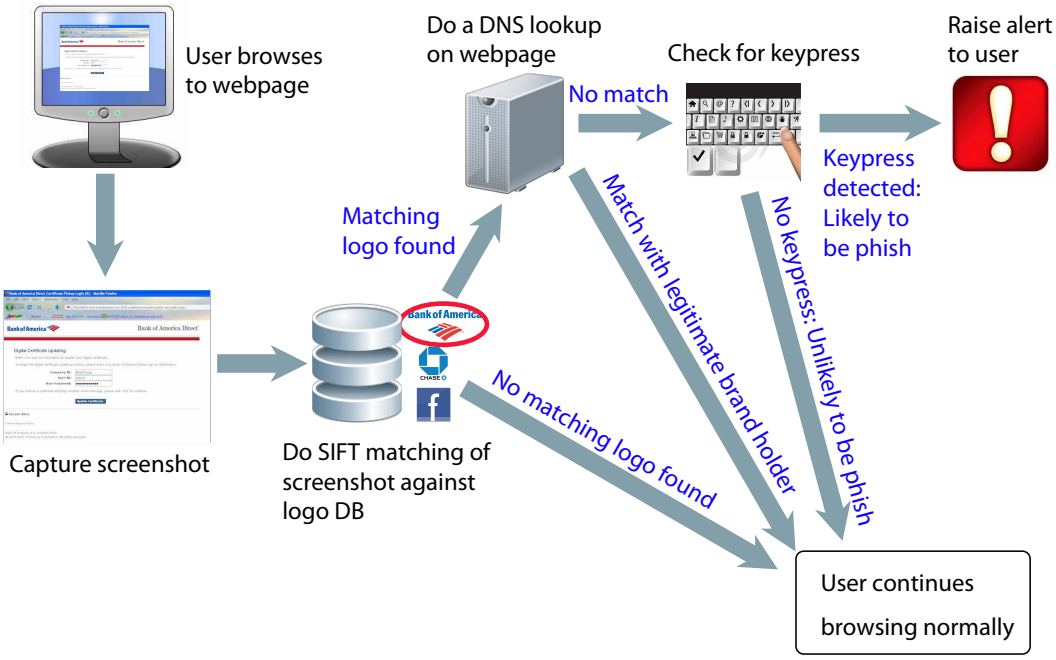


Figure 4.1: Workflow of the Verilogo prototype. As sites are rendered the viewport contents are matched against a logo database using SIFT. Upon a sufficiently strong match we contact the brand-holder for authorization. Absent such authorization, we defer any alert until the user provides keyboard input.

Our prototype, called Verilogo, consists of two components: a server and a Firefox browser plugin. The basic workflow (illustrated in Figure 4.1) mirrors the description in Section 3 and we describe the implementation itself below.

4.1 Prototype implementation

Our prototype uses a matching server spawned in a separate process (this was done for ease of implementation, and could easily be integrated into the browser's address space). It uses a modified version of Robb Hess' SIFT code [15] using the OpenCV 2.0 library for basic vision-related data structures and algorithms. The matching server pre-loads all key features from the logo database (which contains 352 instances in our experiments) and initializes a FIFO for communicating with the browser.

In the browser, we provide a Firefox plug-in that hooks event notifications to determine when a page is loaded and then grabs the entire "viewport" using the existing Firefox "Screengrab" plugin [27]. [1] The resulting bitmap is saved to disk as a PNG file and its location is then communicated to the matching server over their shared FIFO. These operations (in addition to extensive logging for debugging purposes) are run in their own thread context and, while this adds overhead, they do not block the user and generally are overlapped with browser I/O and user reaction time. The browser also records whether the page receives any keystrokes, for use in later filtering (matching latency is generally sufficiently fast that it is not necessary to serialize this step with matching).

When logos are used to indicate that a site represents a brand, they are typically found at the top of the Web page. To optimize this common case, the matching server evaluates the viewport in a series of overlapping strips (starting from the top). By overlapping the strips by N pixels we ensure that any logo of size N can be captured in its entirety in a single strip (in our experiments we use N=100). As N increases overhead decreases, but the latency to find logos on the top of a page is increased. Moreover, logos larger than N may be matched imperfectly.

The matching server also crops the widest possible section of each strip that contains a single color (common on width-limited pages and, by definition, not containing logos). Finally, the server implements SIFT matching against the reference database. If any logo match percentage is above a set threshold, it is reported back to the browser via a complementary FIFO channel (this means that multiple logos appearing on one page

[1] A more sophisticated plug-in implementation could perform incremental screen captures to minimize latency, which our matching server is already designed to handle.

would result in multiple matches). In our implementation we use only a single threshold, but accuracy could be enhanced further by setting per-logo thresholds to reflect their uniqueness.

Finally, upon receiving a signal that a logo has been found, the browser initiates a DNS request to the associated brand holder for an associated BAP record and validates that the hosting IP address is contained within. If it is not, and if a keystroke has been received, then the browser displays an alert.

Moreover, while most anti-phishing alerts are generic ("this page is bad"), we are able to provide semantically meaningful context to allow the user to reason about their situation. In particular, we can communicate that "an alert is being raised because this site is diplaying the Bank of America logo (shown here), but is not authorized to do so by Bank of America. If you are being asked to do something risky, such as provide your Bank of America creditials, it is likely that this site is fraudlent and you should not continue using it." This allows the user to evaluate whether the site really is attempting to represent the Bank of America and act accordingly. Although security dialog "clickthrough" is a challenge facing any security warning, we believe that providing specific and meaningful information rather than general or "jargon-y" warnings is much more helpful in guiding informed user decisions.

In future iterations of our implementation, we will explore the possibility of sending suspicious pages to a blacklist of brandholder to verify as potential phish.

Chapter 4, in full, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Chapter 5

Evaluation

In this section we evaluate our Verilogo prototype, first for performance and then for accuracy.

5.1 Datasets

For our evaluation we consider several datasets. The first, which we call Brand, contains a series of screenshots from legitimate brandholders that feature their logo (e.g., www.bankofamerica.com for Bank of America). We selected twenty-three brands chosen arbitrarily from several lists of commonly phished sites [24, 3, 2] and for each brand gathered ten distinct brand-owned pages containing their logo (230 in total). The second dataset, called Phish, used the same set of brands, but contained up to ten screenshots from known phishing pages for each brand taken from the PhishTank database (219 in total) [1] Both of these datasets represent examples that we desire our matching algorithm to detect (although we did not perform any prior matching or dataset filtering based on matching results).

We also gathered two datasets to represent "typical" Internet pages. One dataset consists of of 183 screenshots gathered from Alexa's Hot URLs list on January 29th, 2010 and the other consists of 254 screenshots from Yahoo's Random Link service. We combined these two datasets together (for a total of 437 pages) to represent a com-

[1] Comerica and M&I Bank did not have ten such examples in the PhishTank database so their phishes are under-represented.

Table 5.1: Summary of the screenshot instances gathered for the Phish, Brand and Hot/Random datasets. We classified screenshots in the Hot/Random dataset by likelihood of keyboard input.

		Input			
Dataset	Total Instances	None	Incidental	Optional	Essential
Phish	219				
Brand	230				
Hot/Random	438	194	182	29	33

bination of both popular pages (Alexa Hot) and a random sample of Web sites (Yahoo Random) – roughly sampling the range of sites that a user might typically encounter. For debugging purposes, we labeled each page by hand with the set of logos we identified on it. We observed that roughly 10% of these pages contained a logo in our database.

Unfortunately, we do not have access to a user activity trace to test our keyboard heuristic, so instead we hand categorized each of the pages in terms of a subjective assessment of the opportunity and likelihood for engaging keyboard input. We classified pages into four categories:

- **No Input**. Pages that have no opportunities for input.
- **Incidental Input**. Pages that allow input, but for features that are incidental to the normal use of the Web page (e.g. search bars on non-search pages).
- **Optional Input**. Pages that allow input and for which the input provides additional functionality that is part of the site's primary use, but for which it is still common to use the site without providing input (e.g., blog pages that contain a comment section, which tend to have more readers than writers).
- **Essential Input**. Pages for which user input is essential to accessing key functionality (e.g., login pages, survey forms, etc).

These datasets are summarized in Table 5.1.

Finally, our logo database includes logos for 166 distinct brands (again taken from lists of commonly phished brands) with 176 distinct instances. There are more instances than brands because some brand holders have multiple versions of logo (either used at the same time, or that have changed over time). Finally, we store two versions

of each logo, a normal one and a color inverted one (for a total of 352 logos). The inverted version is included separately because it is a common stylistic practice to invert the color scheme (positive space to negative space) *and* because this transformation also inverts the direction of the keypoint gradients (and hence a normal logo will not match an inverted one well).

5.2 Performance

We first consider Verilogo's matching performance against our datasets, which reflects the *latency* before an alert is raised. These tests are performed on a HP Desktop Pavilion Elite desktop with a 2.5Ghz Quad-core CPU (only one core was used for our matching task) with 6GB of DRAM (our memory footprint is modest however).

In Figure 5.1 we show the distribution of latencies to resolve the *first* logo on a page (*if* a page truly is attempting to represent a brand holder this first logo is almost always the one of interest) and time to find complete the analysis of the entire page (i.e., approximating a worst case scenario in which a brand logo at the bottom of the page is a successful trust cue in a phishing site).

We consider two datasets separately. The Phish/Brand case captures the time to find a logo on a typical brand site (whether genuine or fraudulent) and captures the expected latency between rendering and notification. The first logo is identified within the first 1.5 seconds 99% of the time and virtually all such pages are scanned in their entirety within 5 seconds (bank pages in particular tend to be short). We note that the Phish/Brand dataset represent the type of pages where latency is most critical. Since the first logo identified is almost always the correct one (since most brand holders place their logo at the top of the page), we expect the measurement of 1.5 seconds to be the most representative of matching performance.

By contrast, the Hot/Random group of pages exhibits a far greater range. This is largely because many of these pages (particularly in the random set), are much longer (some well over 8000 pixels) or wider (some over 1200 pixels) and matching time is roughly proportional to pixel area. Still, the first logo is matched within roughly 6 seconds for 75% of the pages (for which a logo is found) and all matching within 10 seconds

for 75% as well. The tail of the distribution is quite long however and 2-3% of these pages take over a minute to complete both the first match and complete scan. However, these latencies are non-critical since they proceed in parallel with page viewing and do not correspond to those cases used to represent banks and other heavily phished brand holders (it makes little sense for a phisher to try to garner trust by making a user scroll to the bottom of a 8000 pixel page to find the brand logo).

Finally, we remark that SIFT optimization has become a minor industry unto itself. Commercial implementations may offer performance several times greater than that of the Hess implementation used in our prototype. Moreover, recent research has shown that SIFT is easy to parallelize effectively, both across multicore CPUs and on GPUs [34, 11, 30]. For example, the open-source siftGPU implementation can process an 800x600 image in 1/20th of a second [32].

To summarize, on today's processors SIFT-based scanning of Web pages is completely feasible and is poised to become dramatically cheaper still with respect to computational cost, due to increasing parallelism.

5.3 Accuracy

We first consider the accuracy of the SIFT algorithm itself. We focus here on only the Brand and Phish datasets, as these pages are the ones known to definitively contain logos and are the ones that are most essential for our matching algorithm to detect correctly.

Figure 5.2 shows the ROC curves for matching correctness. We consider a false positive to be when our algorithm detects a logo that was not present on the page, and a false negative to be when we fail to find the logo that does appear on the page. We consider the results using two different threshold rules.

Using a single threshold (Figure 3 (a)), the performance is fairly good. With a threshold of 0.15, over 90% of pages are correctly matched and under 3% are matched to the wrong logo (note such mismatches can occur against any of the 352 instances in our logo database and not only for the 23 logos for which we have collected individual pages).

However, many of the false positives are due to a small number of poorly performing logos that have little discriminative power. For example, the Friendster logo and the inverted version of the Facebook logo are very difficult to distinguish as a result of similar words in a sans serif font (e.g. in headers or titles). Indeed, under a reasonable threshold setting, we find that four logos contribute more than 40% of the false positive cases. To mitigate this problem, we implemented a separate threshold for 13 such poorly performing instances (a multiplicative factor of between 1.3 and 2 we selected by hand). Consequently, common pages with no logo present will be less likely to trigger a match (at the potential expense of missing some matches for these instances). The benefit of this optimization can be seen in the "Multiple threshold" version of the graph (Figure 3 (b)) which shows that a 90% true positive rate is now achieved with 1% false positives occuring.

5.4 Keyboard heuristic

However, the broader Internet is filled with pages that may incidentally contain logos in spite of not intentionally representing brand holders (e.g., a news site covering Bank of America). If we only used logo matching and authorization to detect potential phishing sites, then all such pages would incorrectly lead to false alerts being generated.

To get a sense for how many of these pages exist, Figure 5.3 graphs the match rate as a function of the threshold value for the Hot and Random datasets (representing the "typical" kinds of pages that users may encounter on the Internet). It is clear that while for reasonable settings of the threshold most of these pages do not match, there is still a significant number which do. At a threshold of 0.17, 10% of the pages match at least one logo in our database (however, many of these matches are likely to come from the "poorly performing" logos we mentioned earlier).

However, our browser does not necessarily generate an alert for each of these pages, and will only notify the user if they provide keyboard input into the page. Thus Figure 5.3 is a conservative measure of our system's false positive rate and the true false positive rate is the intersection of this set with some function the site content and user behavior. However, evaluating this heuristic is difficult to do directly. As we indicated

earlier, lacking a live user trace, we have used qualitative classifications of user input likelihood to capture different ranges of potential false positive outcomes.

Figure 5.4 illustrates this impact while evaluating the accuracy of the complete system. Each graph provides a ROC curve in which a false positive indicates that an alert would have been generated when it should not have been, and a false negative is a phishing site that was not detected. False negatives are calculated *only* against the Phish dataset (this is purposely pessimistic to test the worst case; in practice phishing pages are a small number of the pages visited) while false positives are calculated against the Hot/Random dataset (which more closely resembles the pages that users will typically visit and thus stresses our keypress heuristic).

In both graphs the lines labeled "Conservative" represent the ROC curves in which a keypress is deemed to have happened for any page not in the "No Input" class (i.e., if there was any way to provide input, then we assume you did so). The "Expected" curve is slightly more liberal, and assumes that keypresses *also* did not occur on pages in the "Incidental Input" category (input possible, but unlikely as with search bars). Finally, the "Optimistic" curve assumes that keypresses also did not occur in the "Optional Input" category (sites like blogs for which input is common, but not necessary) but that users always entered keystrokes in the "Essential Input" class.

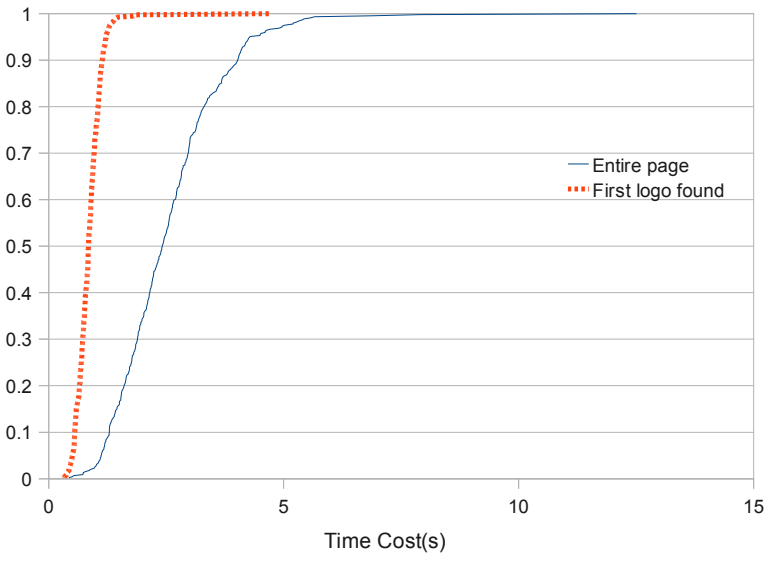
Under the most conservative interpretation false positives can be held under 1% with a true positive rate of 83%. This is further reduced to under 0.5% under either of the more aggresive assumptions. A more significant difference arises when pushing the true positive rate to 90%. Here the conservative assumption produces to an 8% false positive rate, while the expected and optimistic models produce 1.4% and 0.7% respectively. However, we believe that these two curves are more likely to reflect real world use.

Finally, we conducted a small user study to explore the effectiveness of the keypress heuristic when applied to real user browsing behaviors. We asked six participants, all graduate students from the Computer Science department, to install and use a modified version of the Verilogo browser plugin. The modified plugin does not display any alerts or perform any DNS lookups. However, it performs the image matching step and detects if a keypress has been incurred. Using the collected data, we could obtain the set of websites that had a logo detected and incurred a keypress. When we remove all

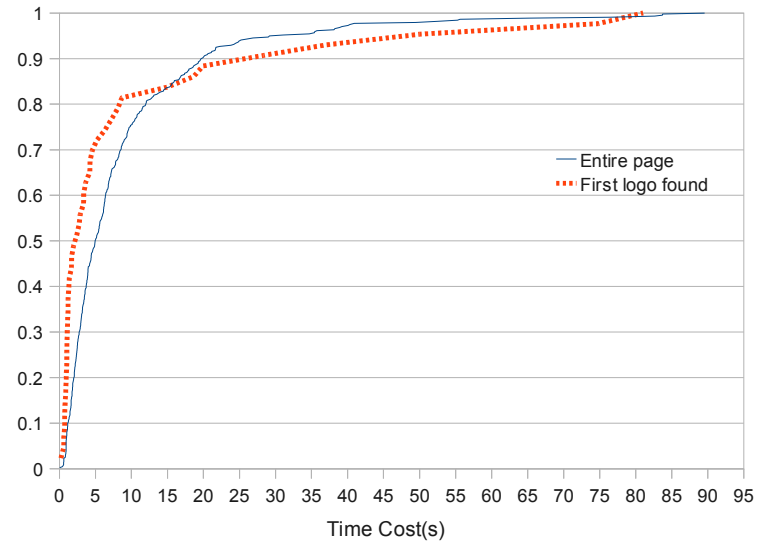
brand holder web pages and phishing pages from this set, we would be left with the set of ”false positive” pages (non-brand holder pages that are not phishing sites, but that would have generated an alert from our system).

We asked the participants to use the plugin for the span of a few hours to a few days. In total, 864 websites were visited and each participant visited an average of 144 websites in the course of the study. We examined the set of websites that both had logos detected by the matching algorithm and had incurred keypresses, and removed all brand holder sites from this set. The remaining websites should have been either phishing pages or false positives . However, there were no such web pages, meaning that we had achieved a false positive rate of 0%. Although the human factors study was performed with only a small number of participants, we believe that these results are encouraging and indicate that our simple keypress heuristic may be an effective one for filtering out non-brand holder, non-phishing sites that contain a brand holder logo.

Chapter 5, in part, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

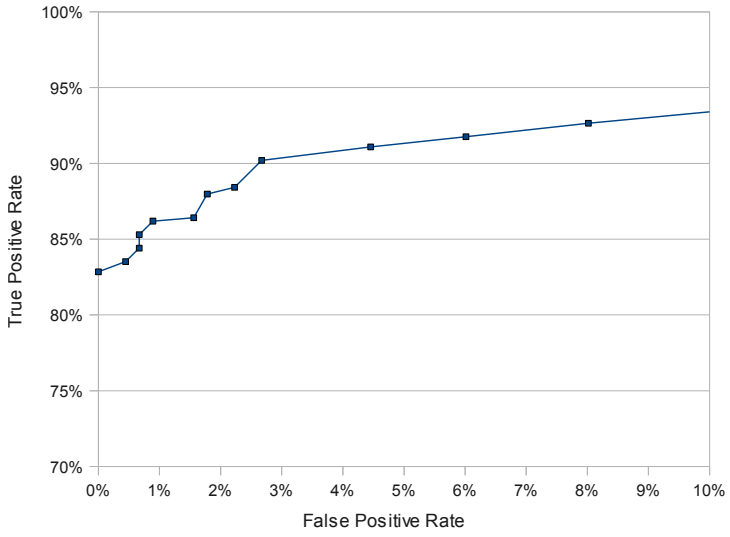


(a) Phish/Brand examples

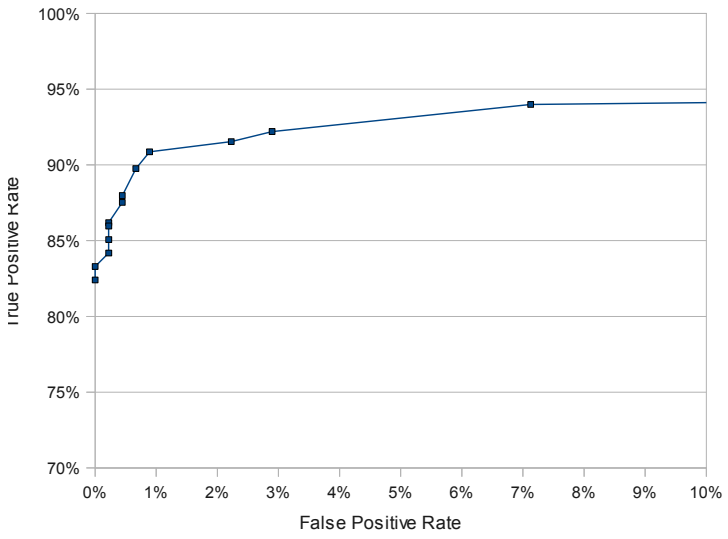


(b) Hot/Random examples

Figure 5.1: Cumulative Distribution Function (CDF) of the time to match the first logo found in a page, and the time to complete analyzing a page.



(a) Single threshold



(b) Multiple threshold

Figure 5.2: Receiver Operating Characteristic (ROC) curve of match performance on the Phish and Brand datasets. The graph on the left uses a single threshold value for all logos while the one on the right increases the threshold for a small number of indistinct logos.

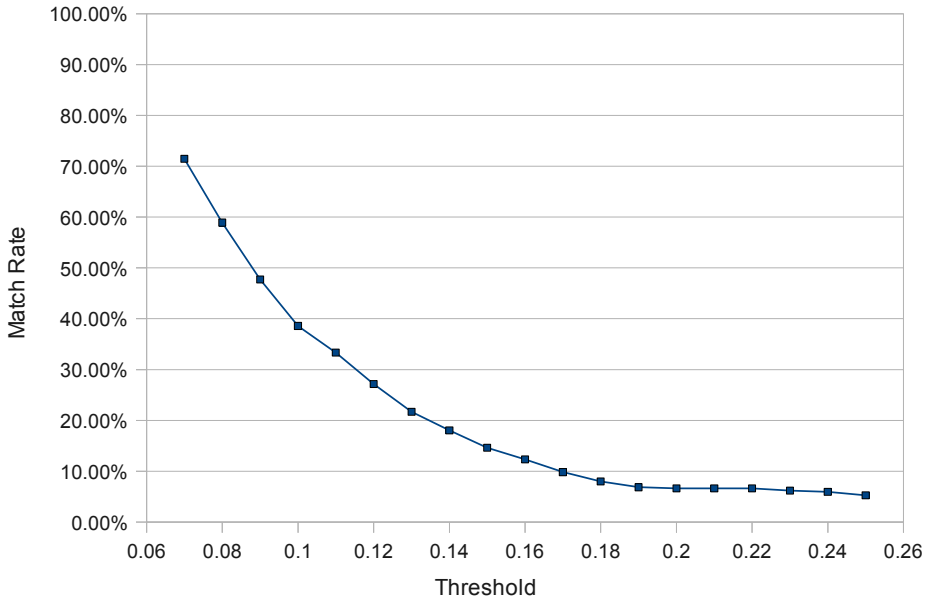
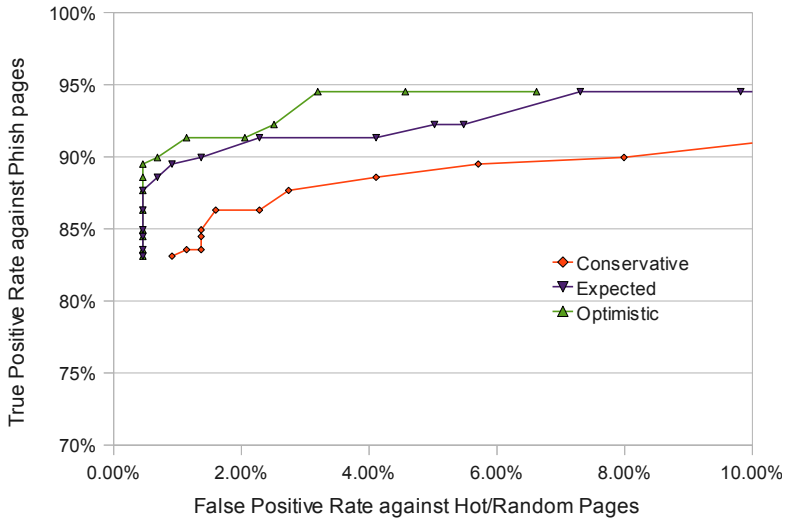
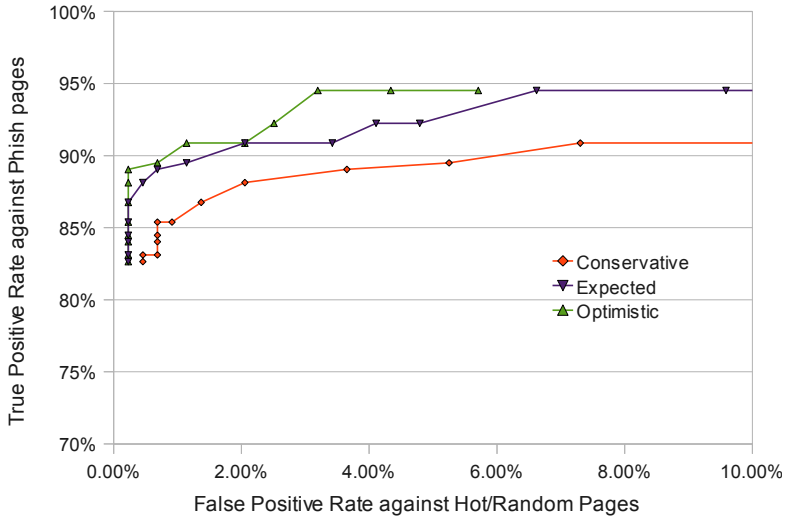


Figure 5.3: Match rate of our 352 entry logo database against pages in the Hot/Random dataset as a function of matching threshold. We use a single threshold for each logo.



(a) Single threshold



(b) Multiple threshold

Figure 5.4: Receiver Operating Characteristic (ROC) curve of overall accuracy on the Phish and Hot/Random dataset. False negatives are calculated against the Phish dataset while false positives are calculated against the Hot/Random dataset. The graph on the left uses a single threshold value for all logos while the one on the right increases the threshold for a small number of indistinct logos.

Chapter 6

Discussion

There are a number of issues that arise from our experience using Verilogo, ranging from innate problems with matching certain kinds of logos to our thoughts about how a sophisticated phisher might try to evade our approach and the privacy concerns introduced by logo authorization. We touch on these briefly here.

6.1 Issues with logo matching

In the course of running our evaluation we observed issues with certain logos and brands that made them a challenge for Verilogo. We discuss these issues below.

Widely distributed logos Some logos, such as Visa's, are widely distributed on legitimate e-commerce sites, which of course require keyboard entry. Because our method (as we have considered it, in isolation) determines whether a page is a phishing page based on whether it includes an unauthorized logo and whether it requests keyboard input, such e-commerce pages will likely register as false positives. With wide distribution, it is unrealistic to expect that Visa will be able to individually track and authorize every page that uses its logo. It may be impossible to secure logos that, by their use, are extremely widely distributed, using techniques such as ours that are designed to enforce the limited distribution of a logo.

Highly variable logos In our study, we found that several of the brands in our study were represented by several variant logos; when our initial logo set was incomplete, this resulted in poor matching behavior. For example, Orkut is represented by a number of different marks, and PayPal has two versions of their logo, one with hollow text and one without. When several different logos represent a brand, all must be included in the database. (This is especially important when a brand changes from an old logo to a new one, as phishing pages could take advantage of the brand equity associated with the old mark even if all legitimate brandholder sites have switched to the new one. However, we argue that popular brand holders are unlikely to change their logos frequently, as this would erode the recognizability of the brand.) A few logos (notably Google's) appeared at widely varying scale. Although SIFT is intended to be scale-invariant, very small versions of logos become pixelated and lose information, leading either to more false positives or more false negatives. However, we believe that, using a relatively small number of logo database entries, we can provide coverage and protection for most of the commonly phished brands. We propose to explore the scalability of our logo database to include brands from smaller organizations, as well as the frequency of required updates in future work.

Figure 6.1: This shows a logo that produces an insufficient number of keypoints to be reliably used for image matching. The left image is the original logo; the right image shows the logo after the feature generation step, with keypoints indicated.

Logos with few keypoints In the course of our evaluation, we encountered several poorly-performing logo instances which often yielded false positives or false negatives.

One cause of poor matching performance seems to be poor discriminatory power in certain logos. The Orange logo seen on the left in Figure 6.1 is one such example — it is a very simplistic, consisting of white text in an orange square. In these situations, the feature generation step may find only a few scale-space extrema, which in turn yields an insufficient number of keypoints that can be used for matching. This leads to poor matching performance for that particular logo. In the case of the Orange logo, feature generation can only discover three keypoints in the logo due to the lack of discernable detail.

One approach that is more suitable to these logos is maximally stable extremal regions (MSERs) [22]. This technique searches for stable "blobs" in an image by applying thresholds and then retrieving a set of extremal regions to use as elements in image matching. An alternative option might simply be removing such logos from the database to minimize potential false positives (since they lack discriminatory power, they are not very useful at detecting phishing sites either).

Text-based logos Another class of logos with poor discriminatory power is the set of logos that consist of text rendered in a commonly used font face. For example, it is difficult for our matching algorithm to distinguish between the Friendster logo and the inverted version of the Facebook logo due to the similarity of the font face and similar letters in the two names (Figure **??**). One potential way to deal with this issue is to supplement image matching with optical character recognition (OCR) to verify against several of the top matching logos. If the similar logos consist of graphical elements rather than text, such extreme similarities may be indicative of infringement issues. Another example of textual regions creating problems in image matching is the "CNN.com" logo. Although the "CNN" portion of the logo is distinctive, the ".com" portion is not. As shown in Figure 6.2, the ".com" part of the logo causes false positive matches due to the occurrence of the ".com" text in the same font face in unrelated web pages. One mitigation technique may be to weight the importance of feature points, as suggested by Frome et. al. [12]. This approach determines weights on each keypoint automatically by statistically calculating the distinctness of that keypoint. More distinct keypoints are assigned a higher weight, and thus points which matched keypoints in the "CNN"

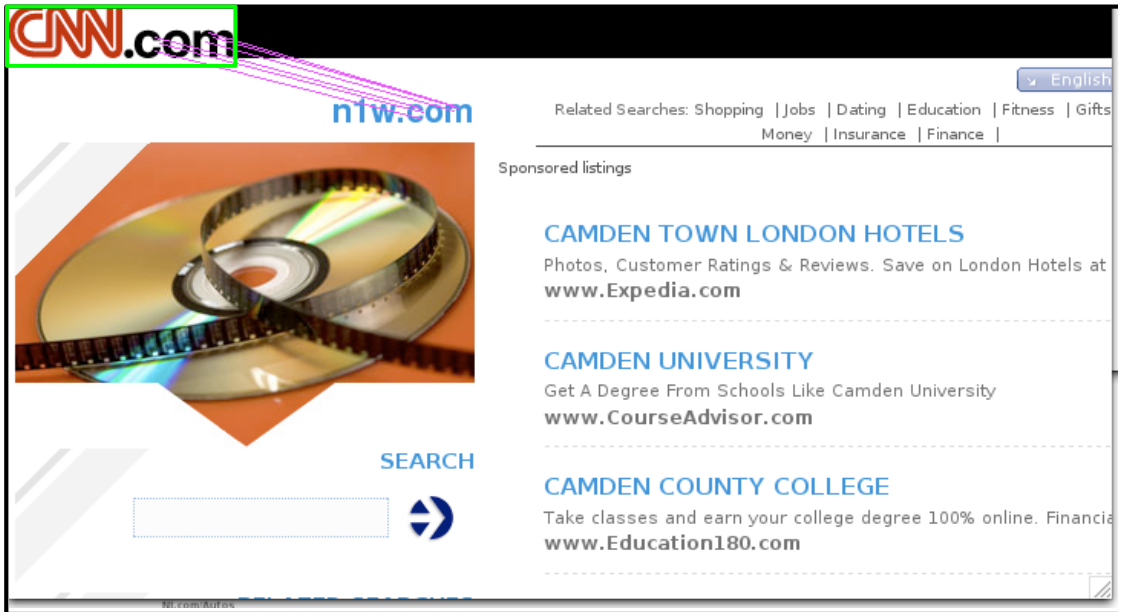


Figure 6.2: The CNN logo is displayed in the green box and a non-CNN screenshot is shown below. The matching feature points between the string portion of the logo and the text in the screenshot triggered a false positive.

portion of the logo would be given more confidence. An even more straightforward solution is to simply identify extraneous logo regions and exclude them competely from the image matching.

6.2 Possible evasion techniques

No defense is foolproof, and ours is no exception. Phishing sites can attempt to evade our system in one of two ways. Either they can set up their pages so that Verilogo cannot find logos on the page, or they can take advantage of our keyboard-entry heuristic to avoid having an alert raised.

An area of concern is whether phishers can transform a logo to the point where the matching algorithm will no longer be able to detect the logo, or omit it completely, but still be able to fool a user into accepting the site as a legitimate brand holder site. Our intuition is that SIFT has proved quite effective at matching images across a range of transformations. Any transformations that would defeat it would alter the logo drasti-

Table 6.1: Summary of the transformations applied to the logos of sixteen brandholder web pages. Using a threshold of 0.15, SIFT failed to correctly detect the logo in six of the screenshots.

Brand	Transformation	Match?
Alliance	30 degree shear x	no
Amazon	removed part of logo and changed font	no
Capital One	6 degree rotate	yes
Chase	3 degree rotate	yes
Citi	removed part of logo and changed perspective	yes
Comerica	changed perspective	yes
eBay	30 degree shear y	yes
Facebook	changed font	no
Google	resized logo	yes
HSBC	removed part of logo	yes
IRS	removed part of logo	no
PayPal	20 degree shear x	no
US Bank	changed perspective	yes
Wachovia	removed part of logo	yes
Wells Fargo	changed font	no
Western Union	10 degree shear x	yes

cally enough to be noticeable to people. Omitting a brand logo completely would likely significantly reduce the effectiveness of a phishing site.

To test our claim that SIFT is tolerant of logo transformations, we collected screenshots of sixteen brands whose logos are in our logo database, and applied simple transformations to the logo portion of the screenshots. Table 6.1 summarizes the transformations applied to the logos. An example of a modified screenshot appears in Figure 6.3.

We performed logo matching on the modified screenshots using a threshold of 0.15, and found that six of the screenshots were not correctly matched. Examining the correctly matched logos, we noted that SIFT appears to be tolerant of rotation, shear up to 10 degrees in X and 30 degrees in Y, changes in perspective, changes in size, and removal of parts of the logo in certain cases. However, SIFT can be defeated by more drastic shear transformations, removing key distinctive portions of the logo, or changing the logo font.

Next, we wanted to determine if modifications to the logos in the brandholder web pages would arouse user suspicion. We conducted a human factors study to find the

answer. For the user study, we used the sixteen modified screenshots, and also collected an additional 49 screenshots belonging to popular brand holders. The study consisted of 11 participants. Participants were graduate and undergraduate students as well as staff from the Computer Science and Engineering departments. We asked participants in the study to rate their familiarity with each of the 65 web pages (the modified web pages were interspersed with the unmodified ones), and whether the web page appeared to be "suspicious-looking" or "phishy". Participants were also asked to explain why a page looked suspicious. The 49 unmodified web pages were included in the study as a control set, in order to examine the effect of priming participants by asking them to look for suspicious web pages.

On average, 19.7% of participants reported unmodified webpages as suspicious, while 51.7% reported modified webpages as suspicious. Interestingly, these numbers changed only marginally (by 1 or 2%) when considering only the set of data where participants reported being moderately to very familiar with the given web page. This is likely due to a combination of brand recognition (participants may notice or be familiar with a brand's logo but may not visit the web page) and aesthetics (even if a user is not familiar with a brand, certain transformations, such as rotation or a drastic shear, may seem aesthetically "wrong"). Another interesting observation is that certain unmodified web pages appeared to participants to be particularly suspicious-looking. For example, 6 out of 11 participants reported Microsoft's "live.com" website as being suspicious. Furthermore, a number of participant responses indicated that poor aesthetics, or what appeared to be an unprofessional design, often contributed to a belief that a page was "phishy".

Overall, it appears that a number of users would be alerted to a phishing attack if the brand logo was modified as described. Furthermore, many of these transformed logos would still be recognized by the image matching algorithm. Thus, the efficacy of an attack that attempted to subvert Verilogo by using simple transformations to modify a brandholder logo would likely be quite limited. We expect that modifying a logo in a way that would both subvert SIFT and almost completely escape user detection is a non-trivial effort.

Another technique that attackers may use to evade logo detection is to mount an

attack on less popular brands whose logos are not in our database, or on webpages that rely on other trust cues (such as favicons or trust seals) rather than logos. We plan to investigate the scalability of our approach with respect to handling larger logo databases and including alternative trust cues, as well as the ease of updating the database to include new trust cues.

We have encountered sparse examples of phishing sites which make use of virtual keyboards or pinpads for login. These techniques would bypass our current mechanism. However, we believe that, first, forcing phishers to rely on virtual keyboards when their legitimate targets do not will reduce the effectiveness of phishing; and, second, that Verilogo can be refined with additional heuristics in anticipation of such attacks. We stress that our keyboard-input heuristic is intentionally a simple and conservative one to allow better evaluation of the underlying scheme.

Phishers might also attempt to separate the "landing" page featuring the logo from the page in which the user is prompted to enter his private information. Since this second page would have to be free of any logos, this should again reduce the effectiveness of the phishing site. It should also be possible to modify Verilogo to track logo-taint across page navigation by click, so that a landing page featuring the Bank of America logo raises a warning when, on the next page, the user initiates keyboard input.

6.3 User privacy

One concern with our proposed DNS-based system for logo authorization is the fact that the DNS queries it uses to determine if the logo appears on a page owned by the brand holder divulge, first, the fact that the particular user has viewed a page at the site being queried about and, second, that this site includes the brandholder's logo. (Arguably the second concern has a positive side, since such reports can facilitate site takedowns.) One solution is to use a DomainKeys-style approach, in which users do not communicate directly with brandholder servers. Alternatively, we believe that cryptographic techniques can be used to perform BAP-record lookups in a privacy-preserving way.

Chapter 6, in part, has been submitted for publication of the material as it may

appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Figure 6.3: The PayPal website is shown here with a modified PayPal logo that was transformed with a 30 degree shear in X.

Chapter 7

Conclusions

In this thesis, we have explored the potential of an image-based system for automatically recognizing branded logos on Web sites for the purposes of fraud detection. In a sense, this capability goes to the essence of how fraud detection works – first a site is evaluated as representing some brand and then it is checked to make sure the representation is unauthorized. In practice, this approach is inherently reactive and inevitably time consuming. We suggest that automated logo matching and authorization is an appealing complement (proactively achieving over 90% accuracy). We've demonstrate a working implementation and shown that it is feasible to integrate near real-time analysis into browsers on today's computer systems (and will only become cheaper since Web pages are not increasing in size as fast as processing power is increasing). Further, we demonstrate that a simple and robust heuristic, deferring alerts until the user has entered keyboard input, is sufficient to limit false positives to between 1 and 2% under reasonable models of when users interact with pages.

Chapter 7, in full, has been submitted for publication of the material as it may appear in the ACM Conference on Computer and Communications Security, 2010. Wang, Ge; Liu, He; Becerra, Sebastian; Wang, Kai; Belongie, Serge; Shacham, Hovav; Savage, Stefan. The thesis author was the primary investigator and author of this material.

Bibliography

[1] Internet fraud, scam and crime statistics - 2009. http://www.consumerfraudreporting.org/internet_scam_statistics.htm, 2009.

[2] Statistics about phishing activity and phishtank usage, december 2009. http://www.phishtank.com/stats/2009/12/?y=2009&=12, 2009.

[3] Top targets for the identified threats. http://www.ghacks.net/wp-content/uploads/2009/12/toptargets.png.jpg, 2009.

[4] S. Afrox and R. Greenstadt. Phishzoo: An auomated web phishing detection approach based on profiling and fuzzy matching. Technical report, Drexel University, 2009.

[5] S. Becerra. Phishingpole: A logo recognition system to detect fraudulent websites. Master's thesis, University of California, San Diego, 2009.

[6] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 1000, Washington, DC, USA, 1997. IEEE Computer Society.

[7] J.-Y. Chen and K.-T. Chen. A robust local feature-based scheme for phishing page detection and discrimination. In *Proceedings of the Web 2.0 Trust Workshop*, 2008.

[8] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell. Client-side defense against web-based identity theft. In *NDSS*, 2004.

[9] R. Dhamija, D. Tygar, and M. Hearst. Why phishing works. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 581–590. ACM Special Interest Group on Computer-Human Interaction, January 2006.

[10] L. Eggert. Global SPF depoyment. Online: https://fit.nokia.com/lars/meter/spf.html, Apr. 2010.

[11] H. Feng, E. Li, Y. Chen, and Y. Zhang. Parallelization and characterization of sift on multi-core systems. In *Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on*, pages 14–23, Sept. 2008.

[12] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2007.

[13] S. Garera, N. Provos, M. Chew, and A. D. Rubin. A framework for detection and measurement of phishing attacks. In *WORM '07: Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 1–8, New York, NY, USA, 2007. ACM.

[14] P. Hallam-Baker. Secure internet letterhead. In T. Roessler and D. Schutzer, editors, *In Proceedings of W3C Workshop on Transparency and Usability of Web Authentication*, Mar. 2006.

[15] R. Hess. SIFT feature detector. `http://web.engr.oregonstate.edu/~hess/`, 2009.

[16] C. Jackson and A. Barth. ForceHTTPS: Protecting high-security web sites from network attacks. In W.-Y. Ma, A. Tomkins, and X. Zhang, editors, *Proceedings of WWW 2008*, pages 525–34. ACM Press, Apr. 2008.

[17] M. Kumar, P. Prakash, M. Gupta, and R. R. Kompella. Phishnet: Predictive blacklisting to detect phishing attacks. In *Proceedings of the IEEE Infocom Conference*, Mar. 2010.

[18] B. Leiba and J. Fenton. DomainKeys Identified Mail (DKIM): Using Digital Signatures for Domain Verification. In *Proceedings of the Fourth Conference on Email and Anti-Spam (CEAS 2007)*, 2007.

[19] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[20] C. Ludl, S. Mcallister, E. Kirda, and C. Kruegel. On the effectiveness of techniques to detect phishing sites. In *DIMVA '07: Proceedings of the 4th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 2007.

[21] J. Ma, L. K. Saul, S. Savage, and G. M. V. lker. Identifying Suspicious URLs: An Application of Large-Scale Online Le arning. In *Proceedings of the International Conference on Machine Learning*, June 2009.

[22] J. Matas and K. Zimmermann. A new class of learnable detectors for categorisation. In *SCIA*, pages 541–550, 2005.

[23] E. Medvet, E. Kirda, and C. Kruegel. Visual-similarity-based phishing detection. In *SecureComm '08: Proceedings of the 4th international conference on Security and privacy in communication netowrks*, pages 1–6, 2008.

[24] T. Moore. Personal communication, 2010.

[25] T. Moore and R. Clayton. Examining the impact of website take-down on phishing. In *eCrime '07: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pages 1–13, New York, NY, USA, 2007. ACM.

[26] P. Moreels and P. Perona. A probabilistic cascade of detectors for individual object recognition. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 426–439, Berlin, Heidelberg, 2008. Springer-Verlag.

[27] A. Mutton. Screngrab! `http://www.screengrab.org`, 2009.

[28] T. Oda, G. Wurster, P. van Oorschot, and A. Somayaji. Soma: Mutual approval for included content in Web pages. In P. Syverson and S. Jha, editors, *Proceedings of CCS 2008*, pages 89–98. ACM Press, Oct. 2008.

[29] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang. An empirical analysis of phshing blacklists. In *Proceedings of the Conference on Email and Anti-Spam (CEAS 2007)*, 2009.

[30] S. N. Sinha, J. michael Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. Technical report, In Workshop on Edge Computing Using New Commodity Architectures, 2006.

[31] M. W. Wong. Sender Authentication: What To Do, 2004.

[32] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007.

[33] C. Yue and H. Wang. Anti-phishing in offense and defense. In *Proceedings of the 2008 Annual Computer Security Applications Conference*, pages 345–354, 2008.

[34] Q. Zhang, Y. Chen, Y. Zhang, and Y. Xu. Sift implementation and optimization for multi-core systems. In *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, pages 1–8, April 2008.

[35] Y. Zhang, J. I. Hong, and L. F. Cranor. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th international conference on World Wide Web (WWW)*, pages 639–648, 2007.