A Hybrid Phish Detection Approach by Identity Discovery and Keywords Retrieval

Guang Xiang
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891, USA
guangx@cs.cmu.edu

Jason I. Hong
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891, USA
jasonh@cs.cmu.edu

ABSTRACT

Phishing is a significant security threat to the Internet, which causes tremendous economic loss every year. In this paper, we proposed a novel hybrid phish detection method based on information extraction (IE) and information retrieval (IR) techniques. The identity-based component of our method detects phishing webpages by directly discovering the inconsistency between their identity and the identity they are imitating. The keywords-retrieval component utilizes IR algorithms exploiting the power of search engines to identify phish. Our method requires no training data, no prior knowledge of phishing signatures and specific implementations, and thus is able to adapt quickly to constantly appearing new phishing patterns. Comprehensive experiments over a diverse spectrum of data sources with 11449 pages show that both components have a low false positive rate and the stacked approach achieves a true positive rate of 90.06% with a false positive rate of 1.95%.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and Protection*; H.3.3 [**Information Systems**]: Information Search and Retrieval; I.2.7 [**Computing Methodologies**]: Natural Language Processing

General Terms

Algorithms, Languages, Security

Keywords

Anti-phishing, named entity recognition, information retrieval

1. INTRODUCTION

Phishing is a form of identity theft, where criminals create fake web sites that masquerade as trustworthy organizations. The goal of phishing is to trick people into giving sensitive information, such as passwords, personal identification numbers, and so on. Recently, the annual Internet crime report of IC3 [15] revealed that Internet crimes had caused a loss of $239.09 million dollars in 2007.

Phishing patterns evolve constantly, and it is usually hard for a detection method to achieve a high true positive rate (TP) while maintaining a low false positive rate (FP). In this paper, we propose a novel hybrid detection method based on IE and IR techniques in an attempt to achieve a good balance between TP and FP. The identity-based detection component of our framework utilizes IR techniques to recognize the identity a webpage claims and captures phish by examining the discrepancy between the claimed identity and its own identity. Named entity recognition (NER) algorithms are used to reduce false positives. This identity-oriented component is aided by a keywords-retrieval component that employs search engines to detect potential phish via searching keywords of significant importance with respect to IR. For instance, a phishing site in Fig.1 claims to be eBay, while actually its true identity is a phishing domain "ovmu98yn1xcy13281mz1.com". Our approach exploits this discrepancy as well as keywords of IR significance from the page ("ebay bid account password forgot") to catch it. To control false positives, we use a domain whitelist and a login form detector to filter good webpages. Experiments over a diverse spectrum of data sources with 11449 pages showed that our approach achieved a true positive rate of 90.06% with a false positive rate of 1.95%.

One major advantage of our method is that it requires no training data, no prior knowledge of phishing signatures and specific implementations, and thus is able to adapt quickly to the constantly appearing new phishing patterns. Traditional blacklist-based method demands an up-to-date phish database to examine new URLs or learn machine learning models, and thus is slow in responding to new phishing attacks. Another property of our approach is that it attacks the TP/FP dilemma by investigating two subcomponents both with low FP and reasonable TP yet focusing on different phishing patterns, and boosting the detection performance via an integrated system.

This paper is organized as follows. Section 2 introduces relevant anti-phishing literature, followed by an overview of our system in section 3. In section 4, we introduce the login form detection module, and subsequently give the detection algorithms in section 5, 6. Experiment setup and result are reported in section 7, and further discussions are given in section 8. Section 9 concludes our work in this paper.

2. RELATED WORK

Recently, anti-phishing has been studied intensively, and a variety of methods have been investigated. Among them, two major camps are blacklist-based methods, which leverage human-verified phishing URLs in an effort to control the

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

false positive rate, and heuristics-based approaches, which utilize HTML or content signatures to discriminate cases and controls. For the latter, machine learning algorithms are usually applied to build classification models over the heuristics to classify new webpages.

In an attempt to exploit visual similarity, Dhamija et al [18] proposed a method named dynamic skins, which uses a unique image for each user-transaction pair in authentication and allows users to visually verify the identity of a remote server by matching the image from the server with its local counterpart.

To capture the patterns in phishing URLs, Ludl et al [19] identified a set of fine-grained heuristics from URLs, and applied a logistic regression model to these URL signatures as well as page rank features, domain-based features and URL-keyword features. Experiments over a repository of 2508 URLs yielded an average TP of 95.8% and FP of 1.2%. Though interesting, this method has high variance in that URLs could be manipulated with little cost, causing the heuristics to fail.

On another frontier, a variety of heuristics have been proposed for phish detection. In [20], the authors came up with a total of 18 properties based on the page structure and achieved using the J48 decision tree algorithm a TP of 83.09% and a FP of 0.43% over a corpus with 4149 good pages and 680 phishing pages. Zhang et al [24] proposed a content-based method using a linear classifier on top of eight features (the TF-IDF heuristic, age of domain, inconsistency of the logo image and domain name, suspicious page URL, suspicious links in the HTML, IP address, number of dots in URL, login forms), achieving 89% TP and 1% FP on 100 phishing URLs and 100 legitimate URLs. The main differences of our work from theirs are that we augmented this keywords-retrieval methodology by an identity-based detection algorithm, obtaining a stacked model with better performance on both TP and FP, and conducted thorough evaluation on a much larger and richer corpus.

In another heuristics-based work, Pan et al [21] proposed a method aiming at extracting the webpage identity from selected DOM properties (such as the page title, meta description field, etc.) via the χ^2 test, and compiled based upon the extracted identity a list of features. With the extracted identity, a support vector machines (SVM) model was trained on 50 phishing and 50 authentic pages, achieving an average FP of about 12% and over 90% TP on a testing set of 50 pages over 7 runs. This is the only work dealing with identity extraction that we know of. However, its assumption that the distribution of the identity words usually deviates from that of the ordinary words is questionable, which is indicated by their high false positive rate. Even in DOM objects, the most frequent term often does not coincide with the web identity. The novelty of our work is that rather than relying on word counts and frequencies, we exploited linguistic features and employed IR and natural language processing (NLP) techniques to find brand domains, which are more robust and effective.

In addition to the research works introduced above, there exist a number of anti-phishing toolbars based on different techniques, many of which exploit blacklists to achieve close-to-zero false positive rate. SpoofGuard [17] extracts phishing signatures via a list of heuristics including seen domains, URL obfuscation, non-standard port numbers, image hashes, etc. A webpage is regarded as phish if the weighted sum of these heuristics exceeds a threshold. NetCraft [1] relies on a central database as well as heuristics such as age of domain, use of IP address, unusual port number, etc., to compute a risk rating for each webpage. Though enjoying a relatively lower false positive rate, these products suffer from poor timeliness and scalability [23] due to the fact that the average life span of phishing sites is usually very short and new phishing patterns appear frequently. Our approach circumvents these problems via a methodology independent of phishing signatures and specific implementations, and thus is able to handle new phishing variants quickly.

3. SYSTEM OVERVIEW

In this work, we define phish to be a webpage satisfying the following criteria

1. It impersonates well-known websites by replicating the whole or part of the target sites, showing high visual similarity to its targets.

2. It is associated with a domain usually unrelated to that of its target website.

3. It has a login form requesting sensitive information.

In some phishing cases, the phishing attack is launched in multiple pages where users need to click a few "continue" buttons before arriving at a phishing page with login forms. Though absent in the beginning, login forms will appear eventually due to the nature of phishing activity.

Our hybrid detection approach exploits a few properties and common practices of website design:

1. Website brand names usually appear in a certain parts of a webpage such as title, copyright field, etc, which renders the website identity searchable and recognizable. For example, term "eBay" appears in many places of its login page (Fig.1), as highlighted by red circles.

2. The universal practice of synchronizing the brand with a domain name lends legitimacy to the strategy of matching textual brand name with domain keyword to determine if a domain truly points to the website the brand refers to. The domain keyword is the segment in the domain representing the brand name, which is usually the non-country code second-level domain such as "Paypal" for "paypal.com" or third-level domain.

3. Phishing webpages are much less likely to be crawled and indexed by major search engines than their legitimate counterparts due to their short-lived nature and few in-coming links.

4. A phishing site usually provides a login form to request sensitive user information, which alone could serve as a feature in classifying webpages.

Our hybrid approach consists of an identity-based detection component and a keywords-retrieval detection component (Fig.2), both manipulating the DOM after the webpage has been rendered in Internet Explorer to get around intentional obfuscations. The former relies on identity recognition to find the domain of the page's declared identity, and examines the legitimacy of the webpage by comparing this extracted domain with its own domain via executing

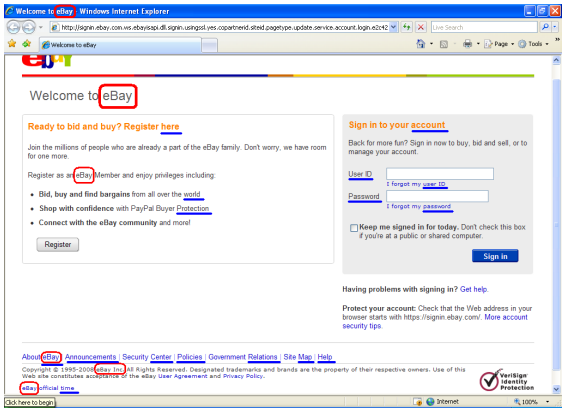


Figure 1: An eBay phishing page with brand name circled in red and words missing ensuing punctuations underlined in blue. The identity-based detection component (Fig.2) executes on search engines query *site:ebay.com "ovmu98yn1xcy13281mz1.com"* and detects the phish by zero search result.

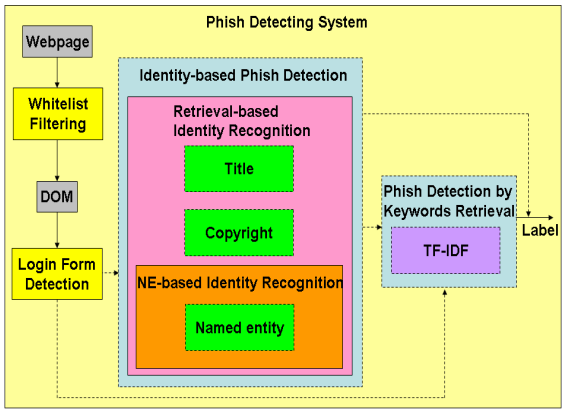


Figure 2: System architecture. The system has an identity-based detection component and a keywords-retrieval detection component.

query of the form *site:declared_brand_domain "page_domain"* in search engines. The two domains in the query are deemed as pointing to the same identity if searching returns results. We do not directly match two domain strings in that some closely related domains (e.g., company affiliations) are literally different (such as "blogger.com" and "blogspot.com"). Using the "site" operator thus reduces such false positives. For the phishing example in Fig.1, our retrieval-based identity recognition module finds the brand domain "ebay.com" based on the brand name "eBay" in the title and copyright field. It then executes on search engines the query *site:ebay.com "ovmu98yn1xcy13281mz1.com"*, and finds no result, indicating the webpage under examination is probably a phish. The NE identity recognition module augments the retrieval-based one in cases where brand names are absent in title and copyright field to control false positives.

Leveraging property 3 above, the keywords-retrieval component, a variant of CANTINA, first identifies from the page content and meta keywords/description tags a set of top ranking keywords using the well-known TF-IDF scoring function, and searches in search engines a query composed of top keywords plus the page domain keyword. Though not all search engines support meta tags, their content still sometimes subsumes valuable information such as the website's identity. A webpage is regarded as a good page if the page domain appears in the top N search results. This augments the identity-based method in the scenario where website identites are missing, thus leading to an improved true positive rate. This keywords-retrieval method is based on the work [24], which also explored the TF-IDF metric.

Domain whitelists and a login form detector sit in the front end of the system, filtering safe pages from further examination. Though this strategy tends to ignore the first a few pages with no login forms of a multi-page attack, it is still able to catch the phish as long as login forms appear eventually, and no harm has been done to the users so far.

4. LOGIN FORM DETECTION

In this section, we present a heuristics-based algorithm using the HTML DOM to identify login forms. Typically, the presence of login forms on a page is characterized by three properties, i.e., FORM tags, INPUT tags and login keywords such as password, PIN, etc. INPUT fields are usually to hold user input and login keywords guarantee that we are actually facing a login form rather than other types of forms such as the common search form. We compiled 42 login keywords to allow flexibility in detecting various patterns such as "passcode", "customer number", etc.

Due to phishing and other inadvertent behaviors, a login form does not always satisfy all three properties above, and to cope with such variations, we designed the following algorithm to declare the existence of a login form.

1. We first handle the regular case in which form tags, input tags and login keywords all appear in the DOM. Login keywords are searched in the text nodes as well as the alt and title attributes of element nodes of the subtree rooted at the form node. Return **true** if all three are found.

2. We then handle the case where form and input tags are found, but login-related keywords exist outside the subtree rooted at the form node f. First, examine whether the form f is a search form by searching keyword "search" in the same scope as in step 1. If f is not a search form, traverse the DOM tree up for 2 levels starting from f to ancestor node n, and search login keywords under subtree rooted at n in the same scope as in step 1. Return **true** if a match is found.

3. This branch captures the phishing pattern in which forms and inputs are detected, but phishers put login keywords in images and refrain from using text to avoid being detected. Check the subtree rooted at f for text and images, and return **true** if no text is found and only images exist.

4. This branch handles the case where phishers only use input fields and leave out form tags on purpose. Search login keywords and image patterns in a similar fashion as above, but in the scope of the whole DOM tree r, and return proper results.

The heuristics in this algorithm may flag a form as a login form when it actually is not. However, this slightly larger

coverage on one hand helps prevent falsely filtering a phishing page prior to the content analysis stage, and on the other still removes a vast majority of pages with no login forms from consideration, thus reducing false positives and accelerating the detection process.

5. IDENTITY-BASED PHISH DETECTION

Two algorithms exploiting website brand identities are given in this section. The basic idea is to first locate entity names in DOM text nodes or attributes of element nodes that are most likely to represent the site brand name, then find domains for those names via searching, and compare the matching domains with the page domain to find identity inconsistency via a strategy defined in section 3. As long as one matching domain is found to be related to the page domain, we classify the webpage as "good". If no matching domains are found, the classification defaults to "good". Both attempt to reduce false positives.

5.1 Retrieval-based Identity Recognition

Our phish detection algorithm in this section solves identity name recognition and name-to-domain translation together in one step via heuristics-aided search, using a page's title and copyright field since they usually contain the site brand name (either its own or the one it is impersonating).

Before delving into details, we give some notational conventions first. Let q denote a search query, w denote a word from q, W_s denote a set of stopwords[1], d denote a domain keyword, ac denote an acronym (defined below) from q and L is a specified minimum length[2].

In light of the first two properties introduced in the beginning of section 3, we find candidate brand domains by searching on major search engines important page fields like title and copyright field, hoping domains corresponding to the brand name in the title/copyright to be returned. We compare the domain of each search result URL with the terms in the search query to find a match[3]. Our algorithm defines four heuristics to evaluate a domain-query match

1. $\exists w \in q, \neg w \in W_s, |w| \geq L, w$ is a substring of d
2. $\exists w \in q, \neg w \in W_s, |d| \geq L, d$ is a substring of w
3. $\exists ac \in q, |ac| \geq L, ac$ is a substring of d
4. $\exists ac \in q, |d| \geq L, d$ is a substring of ac

An acronym is the concatenation of initial letters of a segment in title/copyright, handling the cases where a domain keyword is the combination of initial letters of the brand name[4]. Webpage titles sometimes manifest a certain patterns like "subcategory delimiter category delimiter brand name"[5][6], and to extract acronyms, we define a four-tiered delimiter [7] and split the title iteratively to segments. The copyright field on a webpage typically shows some patterns like "Copyright © 1995-2008 eBay Inc. All Rights Reserved." in Fig.1, and we defined 11 regular expressions targeting different variants to extract the brand name. Some page has more than one copyright field, and we prefer the one with word overlap with the page domain, with keywords like "Inc.", "Ltd.", or simply the last copyright field. Note that sometimes we extract more words than necessary from a copyright field, but still have the brand name in them.

To accurately map a brand name to its domain, we employ Google and Yahoo, two popular commercial search engines, and define two strategies in selecting domains when domain-query matches occur.

- **Strategy I:** *We evaluate domain-query matches among the top* 5 *search results*[8] *of Google and Yahoo. If both search engines have such matches and the domain of the No.1 match from each side coincides, we take it as a candidate domain of the brand corresponding to the query.* If only one search engine has matches, we take the No.1 domain as a candidate brand domain. *Joining the candidate brand domain set also are the first ranked results of both search engines if their domains are identical, regardless of domain-query matches.*
- **Strategy II:** Just take the two branches corresponding to the italicized part of strategy I.

The goal of using two strategies is to investigate whether we have to have both search engines return results with domain-query matches. According to the strategies, at most two candidate brand domains are returned (thus at most two queries), and for each of them, we conduct search using query of the form defined in section 3 with the site operator, and flag a "good" label if either search engine yields results for any query. Otherwise, a phish alarm is fired.

5.2 Named Entity Enhanced Identity Recognition

Website brands often manifest themselves as textual names in places in the page other than title and copyright field, and we can apply NER to identify these names to facilitate phish detection. The focus of this component is mainly to reduce false positives especially in cases where target brand names are absent in title and copyright but are present in other DOM objects. This component first matches recognized entity names with the domain keyword to extract a single name most likely to be the website brand name, uses the search-based mapping algorithm in section 5.1 to obtain its domain and then executes query to identify good sites.

Named entity recognition (NER) is the task of identifying various types of entity names in free text, such as persons, organizations, etc. NER is usually cast as a classification problem under machine learning frameworks [22]

[1]Words that occur very often yet bear little actual meaning such as "the", "of", etc.

[2]The minimum length is defined to be 3 characters here.

[3]In this context, the term "match" means a match between terms in the query and the domain in the query search result according to the four heuristics, not the match where search engines return results for a query.

[4]This usage is not unusual. An example is the brand name "nebraska university federal credit union" whose domain is "nufcu.org", with domain keyword "nufcu".

[5]Two such examples are "Music > Alternative - Mininova" and "Tony Stewart - NASCAR - Yahoo! Sports".

[6]More punctuations may exist in subcategory, category or brand name.

[7]First tier has "|","\:","\>","/"; second has "−"; third has "," and "."; and fourth consists of spaces.

[8]This is a tunable parameter, and we remove noisy URLs like "www.phishtank.com" and "www.millersmiles.co.uk" from the returned list beforehand.

and often explores linguistic features such as part-of-speech tags, affixes (n-grams), etc. In this work, we used the Stanford Named Entity Recognizer [14] to identify website brand names. Stanford NER, a 3-class (organization, person, location) named entity recognizer for English, is a CRF-based information extraction system augmented by Gibbs sampling.

5.2.1 DOM-based End Punctuation Insertion

Formatting tricks via HTML tags ease webpage reviewing, but sometimes omit sentence ending punctuations while not affecting reading, which are of significant importance to the NER task. An example is shown in Fig.1 in which words that should have been followed by an end punctuation are highlighted with thick blue lines. Extracting page content as is tends to produce noisy NER result, and we propose a novel method to attack that problem in this section.

The intuition of our punctuation insertion algorithm is that though various formatting tags are used, non-end punctuations are still necessary to keep the webpage readable, while end punctuations are sometimes omitted. In Fig.1, underlined words miss "." afterwards, while non-end punctuations like "," are all punctuated well. Moreover, a sentence usually ends at the rightmost text node of a DOM subtree. Though occasionally such rightmost text node points to the middle of a sentence, adding a period here does not have a big influence on the following NER step.

Our algorithm exploits the DOM tree and adds a period to the end of either the rightmost text node of a basic block, a text node preceding a BR node, or each text node of a link list structure, when end punctuation is missing. In this context, a basic block is defined to be a subtree composed entirely of anchor nodes and text nodes (except the subtree root), and a link list is a subtree with only anchor nodes or anchor nodes separated by text separator ("|" or "–") such as the 9 anchors on the bottom of Fig.1 starting with "About eBay". Note that both definitions only apply to the DOM tree after processing because otherwise there could be many formatting tags in a subtree like DIV. Link lists are important because there is often a link list on the bottom of a webpage followed by a copyright field, where website brand name appears.

In our algorithm, we first prune the DOM tree by removing non-informative nodes including empty text nodes, SCRIPT nodes, NOSCRIPT nodes, SELECT nodes, STYLE nodes, nodes whose children are all removed, all but the first of contiguous sibling BR nodes and other non-text leaf nodes. We then add a period if there is none to the text node prior to a BR node. Next, we add a period to the end of the page title if necessary and collapse the DOM tree by removing non-text leaf nodes and non-anchor internal nodes that are the only child of their parents. Note that this collapsing step will remove the BR nodes that survive the pruning stage. Collapsing the DOM tree will significantly cut the tree size, and facilitate punctuation addition via basic blocks. In the end, we add a period to the proper positions of a basic block and link list.

The major part of our punctuation insertion method is described formally in algorithms 1 and 2. The procedures of adding periods to link lists and basic blocks and collapsing the DOM tree are omitted. Note that correcting punctuations perfectly is a hard problem, and our approximate algorithm tends to add more punctuations, which is desirable since such redundancy reduces unwanted named entities.

Algorithm 1 AddPunctuationMain

Require: Raw DOM tree r
Ensure: Punctuation-added DOM tree
1: Remove non-informative nodes from r
2: Add "." to text nodes preceding BR nodes if necessary
3: Add "." to title if necessary
4: CollapseTree(r)
5: AddPunctuations(r)

Algorithm 2 AddPunctuations

Require: a subtree root r of the processed DOM tree
1: **if** r is text leaf **then**
2: **if** (r is the last child) && (r not end with punctuation) **then**
3: text$_r$ ← text$_r$+ "."
4: **end if**
5: **else**
6: **if** r is link list **then**
7: AddPunctuations2LinkList(r)
8: **else if** r is basic block **then**
9: AddPunctuations2BasicBlock(r)
10: **else**
11: **for all** child n of r **do**
12: AddPunctuations(n)
13: **end for**
14: **end if**
15: **end if**

5.2.2 Dual-source NE-based Identity Recognition

Our dual-source identity recognition algorithm proceeds by first identifying via NER a list of organization names from the visible content (set 1) and invisible DOM objects including the alt and title attributes of element nodes and the content attribute of meta description tags (set 2), and then applying heuristics to find a single name (or none) that is most likely to be the brand identity from the two sets. Each candidate name in the two sets is split into terms, and its count of matches with the page domain keyword is recorded. A match is found if a term is not a stopword, satisfies a minimum length, and either is a substring of the domain keyword or the other way around. If the acronym of an organization name is identical to the domain keyword, it is also counted as a match. Our heuristics prefer entity names 1) with higher match count, 2) recognized from the page content, 3) with shorter length. This procedure is shown in algorithm 3. After getting a final name from the two sources, we extract its domain and classify the webpage using the query algorithm introduced in section 5.1.

6. KEYWORDS RETRIEVAL FOR PHISH DETECTION

Motivated by the property that phishing webpages are much less likely to be crawled and indexed by major search engines due to their short-lived nature and few in-coming links, we present in this section a method utilizing search engines to detect phish.

In light of the fact that all search engines employ scoring functions to rank matching documents, we should intuitively feed search engines those keywords that are more likely to push intended webpages to top positions in the result list.

Algorithm 3 FindORGIdentityName

Require: domain keyword d, webpage p, DOM tree r
$N_1 \leftarrow$ NERFromContent(p)
$N_2 \leftarrow$ NERFromInvisibleDOMObjects(r)
$\forall n \in N_1, N_2$, break n into terms
$\forall n \in N_1, C_1 \leftarrow$ compute n's terms match count with d
$\forall n \in N_2, C_2 \leftarrow$ compute n's terms match count with d
if non-zero count exists in C_1, C_2 **then**
    if $\max(C_1) \neq \max(C_2)$ **then**
        return the name with highest count, breaking count-ties by choosing the shortest name
    else
        if names with highest match count with d from N_1, N_2 intersect **then**
            return a name in the intersection from N_1 preferring shorter ones
        else
            return a name with highest count from N_1, breaking count-ties by choosing the shortest one
        end if
    end if
else
    return the name with acronym match with d from N_1, or N_2 if N_1 yields none, or none
end if

Toward that end, we adopted the classic TF-IDF metric in ranking candidate query words

$$TF\text{-}IDF(w) = TF(w) \cdot IDF(w)$$

where term frequency TF(w) denotes the number of occurrences of w in the page, and inverse document frequency IDF(w) measures the general importance of w in the whole collection. We used Google as the collection corpus, and estimated the document frequency of a term w by the number of search results on the upper right corner of the result page when searching w in Google. To increase TP and reduce FP, we also put the page domain keyword in the query.

In this algorithm, we also use two search engines, Google and Yahoo, and report a page as phish if neither has the page domain in the top 30 results. The full-blown model integrating the identity-based and retrieval-based detection methods is described in algorithm 4.

7. EXPERIMENTAL EVALUATION

7.1 Evaluation Metric

In our experiment, we adopted the standard true positive rate (also called recall) and false positive rate as the evaluation metrics, in which $p2p, p2n, n2p, n2n$ stand for the number of phishing webpages correctly classified as phish, the number of phishing pages wrongly classified as good pages, the number of legitimate pages wrongly classified as phish (false positive) and the number of legitimate instances correctly classified as legitimate respectively.

$$\text{True positive rate (recall)} = \frac{p2p}{p2p + p2n} \quad (1)$$

$$\text{False positive rate} = \frac{n2p}{n2p + n2n} \quad (2)$$

Algorithm 4 DetectPhish

Require: Webpage p, page domain keyword d, page domain n, white domain list D_w
Ensure: **true** – phish; **false** – good
Parse p
$r \leftarrow$ DOM
FilterByWhiteDomain(n, D_w)
DetectLoginForm(r)
if (n in D_w) || (no login form found) **then**
    return false
else
    $t \leftarrow$ GetTitle(r)
    $cp \leftarrow$ GetCopyright(r)
    terms $\leftarrow$ GetTopTFIDFTerms(r)
    AddPunctuationMain(r)
    $id \leftarrow$ FindORGIdentityName(d, p, r)
    $pred \leftarrow$ DetectByIdentity(d, t, cp, id)
    $pred \leftarrow pred$ || DetectByIFIDF(d, terms)
    return $pred$
end if

7.2 Data and Usage

Phishing sites are usually ephemeral, and most pages won't last more than a few days. To fully study our approach over a larger corpus, we downloaded the phishing pages when they were still alive and conducted experiment in an offline mode. To get around phishing obfuscations, our downloader employed Internet Explorer to render webpages and execute Javascript, and thus the DOM of the downloaded copy corresponded to the genuine page content. Images were downloaded for CANTINA.

White domains are good domains verified by authorities, and serve as an effective way in reducing false positives and speeding up detection. We collected such domains from three sources. First, Google safe browsing provided a whitelist of [2] 2770 domains by mid September of 2008, and we obtained a total of 2682 unique domains after removing duplicates. Second, millersmiles [3] maintains an archive of common spam targets like Paypal, and we extracted 424 unique domains out of a total of 732 entries after mapping organization names to domains and removing duplicates. Moreover, we also utilized an online white domain service [4], which performs DNS lookup to determine if a query domain is on the whitelist. Like any other whitelists, this online database's coverage is rather limited, and out of all the 3543 good URLs we have, only 480 appear on it.

Our webpage collection consists of phishing cases from one source, and good webpages from six sources. To eliminate the influence of language heterogeneity on our text-oriented methods, we only downloaded English webpages.

For phishing instances, we used the XML feed of Phishtank [5], a free community-based anti-phishing site with 28,953 accounts [6] so far. Web users can submit suspected phish, which are then verified via a simple threshold voting mechanism. Genuine phishing URLs are added into a downloadable blacklist after verification, and until October 28, 2008, Phishtank has 350,000 verified phishes after it was launched two years ago. We started downloading the feed in early May of 2008 and manually examined the downloaded webpages to remove legitimate cases, 404 errors, and other types of noisy pages, collecting a total of 7906 phishing webpages during a five-month period.

Good pages came from the following six sources. Alexa.com maintains a top 100 website list for a variety of languages, and we crawled the homepages of the top 100 English sites to a limited depth, collecting 1039 good webpages in this category. To introduce webpages with login forms into our data set, we downloaded 961 login pages, utilizing Google's inurl operator and searching for pages with keywords such as "signin" and "login" in the URL. Although not every page in this category contains an actual login form, there is guarantee that all of these URLs point to legitimate websites. 3Sharp [16] released a public report on anti-phishing toolbar evaluation in 2006, and we downloaded 101 good English pages out of the 500 provided in the report that still existed at the time of downloading. Moreover, we went to Yahoo directory's bank category [7], crawling the bank homepages for a varying number of steps within the same domains and collecting 988 bank pages. Likewise, we conducted crawling on other categories [8, 9, 10, 11, 12, 13] of Yahoo directory including US bank, credit union, online escrow services, travel agencies, real estates and financial services, and gathered 371 webpages. We name this data set "*Yahoo misc pages*" for reference convenience. To test the robustness of our methods, we manually chose 83 login pages of popular phishing target sites, such as eBay, etc. We call this data set "*prominent pages*". Note that none of the other five categories has overlap with URLs in this set, rendering this category independent of others.

In our evaluation, we applied our algorithms to the whole corpus and reported the result statistics.

7.3 Experiment Result

7.3.1 Detecting Login Forms

As shown in Table 1, we successfully detected 99.82% phishing pages with login forms, and filtered a significant percent of good pages from other categories. For the remaining 0.18% (14 in absolute number) phishing pages, they either do not have a login form (very rare in our phish corpus), use login keywords not in our list such as "serial key", or organize the form/input tags in a way our method misses. Note that a lot of webpages in the login category do not have login forms. Pages with keywords like "login" in URLs do not necessarily have login forms.

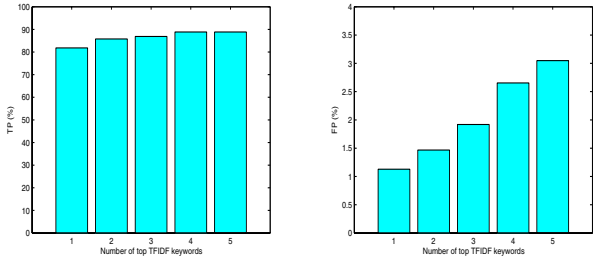
7.3.2 Phish Detection by Keywords-Retrieval

In this section, we report the performance of the keywords-retrieval component, varying the number of top keywords.

Examining both graphs in Fig.3 reveals that throwing more words with top TF-IDF scores in the query may bring up irrelevant result pages that on one hand increase TP while on the other hurt FP. This is an interesting observation contradictive to the thought that more relevant query words will help find the intended webpages more effectively. The secret sauce of Google and Yahoo has not been published, and considering the fact that false positive is usually weighed more heavily in industry, we took only the No.1 TF-IDF word with the domain keyword in building queries in other experiments of this paper.

7.3.3 Identity-based Detection under Strategy I

The effectiveness of each individual module and their combination is interesting to explore, and in this section, we experimented with five approaches, i.e., detection by 1) title, 2) copyright, 3) TF-IDF, 4) title + copyright + NE, and 5) a full-blown method with a combination of the four. Among them, four approaches except the pure TF-IDF one used strategy I (section 5.1) in matching domains of query search result URLs with the query. The TF-IDF-based method does not perform domain-query match.



(a) True positive rate (b) False positive rate

Figure 3: Performance of the keywords-retrieval detection method (section 6). Both TP and FP increase monotonically as the number of TF-IDF-ranked keywords grows from 1 to 5. TP on the left has a minimum of 81.80% and tops at 88.86%. FP on the right ranges from 1.13% to 3.05%. The priority of FP suggests using fewer TF-IDF-ranked keywords for this detection algorithm.

Notice that our NE-based detection algorithm (section 5.2) was only used as an auxiliary module to the identity-based component to reduce false positives, and thus was not tested individually.

A quick glance at the result in Table 2 reveals that all individual detection algorithms have low FP ($< 1.5\%$).

Also shown in Table 2, the identity-retrieval method using title and copyright captured 78.03% and 54.33% phish respectively. About 21.97% and 45.67% phish were missed mainly in cases where either no identity name was found (predict "good" by default) or the page domain was on the whitelist. The latter was caused by phisher hacking into legal domains and uploading phishing sites. Moreover, the 1.38% and 1.41% FP were mostly due to the absence of brand names in the title and copyright field, leading to false domain-query matches and zero result when executing query of the form *site:declared_brand_domain "page_domain"*.

Another cause of false positives was that some extracted domain did not truly represent the intended brand, even if the true brand name appeared in the title/copyright field and a domain-query match was found. An example page of this with URL `http://www.fbandt.com/atmSearch.php` has title "firstbank & trust", while matching the title search results with the title returns domain "firstbank.com", pointing to a different entity.

To examine the effectiveness of domain whitelist and login form filtering, we tested the identity retrieval based detection method using title only in two experiments, with no whitelist and login form detection respectively. Result in Table 2 suggests that though these two filtering steps have no dramatic impact on the TP, applying whitelist does improve the FP slightly (from 1.92% to 1.38%) and skipping login form filtering hurts FP significantly, which plummeted from 1.38% to 3.81%.

Table 1: Statistics of login form detection. 99.82% phishing pages with login forms were successfully detected.

Corpus	Phishtank	Alexa	Login pages	3Sharp	Banks	Yahoo misc	Prominent
#total pages	7906	1039	961	101	988	371	83
#pages detected with login forms	7892	318	639	35	234	98	76

Table 2: Performance of all methods under strategy I. The use of a whitelist degrades the TP of detection-by-title method by 1.15%, due to the activity of planting phishing pages into legal domains. Login form filtering significantly reduces the FP of the detection-by-title method from 3.81% to 1.38%. The combination of Title+Copyright+NE+TF-IDF boosts the TP to 93.31% with a FP of 2.26%. The keywords-retrieval detection method uses the word with No.1 TF-IDF score plus domain keyword.

	Title	Copyright	TF-IDF	Title+Copyright+NE	Title+Copyright+NE+TF-IDF
TP(%)	78.03	54.33	81.80	82.90	93.31
FP(%)	1.38	1.41	1.13	1.38	2.26
	Title only with no domain whitelist			Title only with no login form detection	
TP(%)	79.18			78.09	
FP(%)	1.92			3.81	

Another observation is that title seems to be more effective than the copyright field in directly delivering brand-related information. One reason is that copyright field sometimes gives the name of a parent organization offering a variety of services or products, and the page domain points to one of them, leading to possible false positive when the parent organization domain does not refer on their site to the service or product domain name.

Keywords-retrieval detection outperformed identity-based detection with title and copyright in both metrics, demonstrating the power of commercial search engines in their crawling breadth and document ranking capability.

Enhancing title/copyright with identity NER pushed the TP up to 82.90%. Interestingly, this enhancement kept FP the same as detection by title alone (lower than the 1.41% FP of copyright-based detection), suggesting that the NE-based detection algorithm correctly removed a certain false positives caused by copyright-based detection.

The synthesis of all four algorithms boosts the TP to 93.31%, with a low FP of 2.26%, which suggests that the phish captured by the four methods do not entirely overlap. Though the types of false positives each individual module suffers from are hard to perfectly specified by pure analysis, the stacking strategy will lead to a combined model with low FP as long as each component has reasonably low FP.

7.3.4 Identity-based Detection across Strategies

Besides the individual detection modules, the efficacy of the strategies (specified in section 5.1) in selecting candidate brand domains upon the occurrence of domain-query match is also worth exploring, and we report the evaluation for that purpose in this section. Table 3 shows the experiment result of all detection methods under two strategies. Note that the keywords-retrieval detection method does not involve domain selection for the website brand name and thus shows the same performance under both strategies.

Across strategies, the TPs of title-based detection method were tremendously different, with 78.03% under strategy I and 57.24% under strategy II, and the corresponding FPs also dropped from 1.38% to 0.40%. Considering the different sizes of the phish (7906) and legitimate (3543) corpus in our experiment, these statistics suggest that even if only a single search engine returns top domains with term match with the query, it is still beneficial to take those domains as corresponding to the true brand name since they were able to catch a significant number of phish (over 20% or 1580 pages) at the cost of limited degradation on FP (around 1% or 35 pages). The performance of the full identity-based detection method (Title+Copyright+NE) also confirms this by lifting the TP from 68.23% to 82.90%, with 0.45% decline in FP, suggesting the effectiveness of search engines in discovering brand domains. Another insight is that Google and Yahoo may use different ranking and crawling algorithms, and it is desirable to adopt both for phish detection.

The TPs of the detection-by-copyright approach almost remained identical across two strategies (54.33% vs 54.15%), delivering the message that copyright field is usually more stable for website identity extraction, which makes perfect sense since the purpose of copyright field is to show website brand names while the tile could express any information and thus is much noisier.

Similar to the experiment in the previous section, a stacked hybrid model of four algorithms achieved the highest TP at 90.06% under strategy II, significantly better than each of the individual method, with a low FP of 1.95%.

7.3.5 Evaluation with Other TF-IDF Approaches

In [24], Zhang et al proposed CANTINA, a content-based method, which performed competitively in their experiment against two state-of-the-art toolbars, SpoofGuard and Netcraft. We implemented an offline version of CANTINA, and evaluated our algorithms with CANTINA on the same corpus.

Table 4 shows that the TPs of our algorithms were comparable to CANTINA, while the FPs were much better (2.26%/ 1.95% vs 5.98%). Four hypothesis tests were conducted comparing the TP/FP of our methods under each strategy with CANTINA, all with the null hypothesis hypothesizing equal performance while the alternative hypothesis favoring our method. Table 4 reveals that all but one case are statistically significant (marked by $*$) with strong evidence in favor of our detection algorithms.

Although phishing signatures constantly evolve, the conclusion from [24] still carries and our experiment results suggest that our proposed algorithms are at least as good as, if not better than, the state-of-the-art anti-phishing toolbars.

Table 3: Performance of all methods across strategies. An integrated Title + Copyright + NE + TF-IDF boosts the TP significantly under both strategies. The keywords-retrieval detection method uses the term with No.1 TF-IDF score plus domain keyword.

	Title	Copyright	TF-IDF	Title+Copyright+NE	Title+Copyright+NE+TF-IDF
TP(%), strategy I	78.03	54.33	81.80	82.90	93.31
TP(%), strategy II	57.24	54.15	81.80	68.23	90.06
	Title	Copyright	TF-IDF	Title+Copyright+NE	Title+Copyright+NE+TF-IDF
FP(%), strategy I	1.38	1.41	1.13	1.38	2.26
FP(%), strategy II	0.40	0.90	1.13	0.93	1.95

Table 4: Performance of the full-blown model (Title+Copyright+NE+TF-IDF) under two strategies vs CANTINA. Our algorithms perform comparably with CANTINA in terms of TP, while far outperform it on FP (2.26%/1.95% vs 5.98%). Hypothesis tests compare our methods against CANTINA for each metric under each strategy, with statistically significant results marked by $*$.

	Strategy I	Strategy II	CANTINA
TP(%)	93.31	90.06	91.40
FP(%)	2.26	1.95	5.98
p-value (TP%)	$< 1.0e$-5 $(*)$	0.998	
p-value (FP%)	$\ll 1.0e$-5 $(*)$	$\ll 1.0e$-5 $(*)$	

8. DISCUSSION

8.1 Further Examination on the Performance

In our experiment, part of the false negatives was due to phisher hacking into legal domains, which falsely triggered the whitelist filter. This could be alleviated somewhat by examining the webpage using our algorithms even if domain whitelist yields a match, which will however inadvertently raise the FP a little bit as a side effect. Alternatively, we could use a small whitelist containing only well-known phishing target sites that are possibly hard to break by phishers.

On the other hand, false positives were mainly caused by the following factors. First, the webpage title and copyright do not contain the brand name. Under this case, searching title/copyright on search engines may still return URLs whose domain keywords match a certain terms in the title/copyright, leading to false brand domains and thus false positives. Second, the keywords-retrieval algorithm utilizing the TF-IDF metric does not return the desired domain name in top result entries, possibly due to non-optimal choice of query words even if the TF-IDF function is used. For the former, the keywords-retrieval method can help, while for the latter, the identity-based approach may fill the void. These two complementary components when stacked together are able to boost the TP higher while maintain a low FP.

One major concern about the performance, however, is the time complexity. The full-blown model relies on searching via two search engines to extract brand domains and detect phish, and consecutively querying search engines negatively impacts the running time. Therefore, properly caching search results is of paramount importance to our method, especially for the identity-based detection component. However, searching actually may not have significant impact on our hybrid framework mainly thanks to the following properties. With the increasingly wide use of phish toolkits, highly similar or even identical phishing pages are very common, and though sometimes page titles may vary from case to case, copyright field is usually more homogeneous, especially within the same website. In addition, the final query using the site operator only deals with domains, and is likely to be expedited significantly with caching. Therefore, we expect the cache hit rate to be acceptable. As to classifying good webpages, our system does not suffer from the query traffic problem as much because the vast majority of good pages do not have login forms and thus will be filtered before fed into our detection algorithm.

One potential weakness is that querying with "site" operator may not yield results when legitimate sites use IP addresses. However, this usage is very rare for good sites, and we only see two such cases in our corpus. Moreover, this can be fixed by performing reverse IP lookups to find the domains that resolve to an IP.

In the proposed detection framework, phishers cannot influence the false positives, and a simple yet effective method to further cut FP of the identity-based detection component is through collaboration with web users by educating them to design their websites in a way that the site brand name is laid in a visible and easily detectable place. Ideally, if every website does this, the FP of our identity-based detection component would be close to zero, compared with the 1.38% and 0.4% of the title-based method, and 1.41% and 0.9% of the copyright-based method under two strategies (Table 3). Under this ideal scenario, we could also improve the TP of the identity-based component by imposing stricter constraints to avoid unintended search matches.

In this paper, we only investigated the textual objects of the DOM, and even so, we were able to achieve 90.06% TP and 1.95% FP over a large corpus. Aided by other well-performing features beyond text, we have confidence that our algorithms could improve further on both measures.

8.2 How Phishers could Respond

One thing that phishers could do to evade our detection is to remove from the DOM the brand names of the target websites they are trying to impersonate. However, although this trick might get around the identity-based detection in our framework, the keywords-retrieval approach is still able to catch it. Phishers could go more extreme by removing all textual components and using only images on the webpage. This is a hard case. However, we really doubt if any legitimate entity would design their website this way (especially the login page), and such webpages look very suspicious even without classification. Moreover, the domain of such pages offers clue and the keywords-retrieval method may still work.

Phishers may try to attack our identity-based component by injecting to the DOM a large number of noisy entity

names with tiny (or user-invisible) fonts or background color in hope of paralyzing our system. Such attacks can be circumvented in that text with unusually small fonts or background color is not hard to detect and we could simply remove it during preprocessing. Moreover, our NE-based module searches the site brand name by comparing terms in candidate entity names with the page domain keyword, and those injected random names will not match, thus leading to no recognized brand names and no extra query traffic. As long as we seek target brand names in visible fields such as title and copyright, attackers cannot spoof our method by putting malicious domains in such fields.

Another trick phishers may adopt is to add phishing domains on an posting description of the target site such as eBay, fooling querying via the site operator to return results and thus possibly penetrating our system. We can defeat such attempts by restricting querying to official postings only and filtering matching results from user-added posts.

9. CONCLUSIONS

In this paper, we presented the design and evaluation of a hybrid phish detection method with an identity-based detection component and a keywords-retrieval detection component. The former runs by discovering the inconsistency between a page's true identity and its claimed identity, while the latter employs well-formulated keywords from the DOM and exploits search engines' crawling, indexing and ranking properties to detect phish. Experimental evaluation over a corpus of 11449 pages in 7 categories demonstrated the effectiveness of our approach, which achieved a true positive rate of 90.06% with a false positive rate of 1.95%.

Not requiring existing phishing signatures and training data, our hybrid approach is agile in adapting to constantly evolving phish patterns and thus is robust over time.

10. ACKNOWLEDGMENTS

Thanks to the members of the Supporting Trust Decisions project for their feedback. This work was supported in part by the National Science Foundation under grant CCF-0524189 ("Supporting Trust Decisions") and by Army Research Office grant DAAD19-02-1-0389 ("Perpetually Available and Secure Information Systems"). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation or the U.S. government.

11. REFERENCES

[1] http://toolbar.netcraft.com/.

[2] http://sb.google.com/safebrowsing/update?version=goog-white-domain:1:1.

[3] http://www.millersmiles.co.uk/scams.php.

[4] http://www.uribl.com.

[5] http://data.phishtank.com/data/online-valid/.

[6] http://www.phishtank.com/stats.php.

[7] http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Banking/Banks/.

[8] http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Banking/Banks/By_Region/U_S__States/.

[9] http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Banking/Credit_Unions/.

[10] http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Online_Escrow_Services/.

[11] http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Business_Opportunities/Travel_Agencies/.

[12] http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Business_Opportunities/Investment_Opportunities/Real_Estate/.

[13] http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Business_Opportunities/Financial_Services/.

[14] Stanford named entity recognizer (version 1.1). http://nlp.stanford.edu/software/CRF-NER.shtml.

[15] The 2007 internet crime report. 2007. The Internet Crime Complaint Center (IC3). http://www.ic3.gov/media/annualreport/2007_IC3Report.pdf.

[16] 3sharp report. Gone phishing: Evaluating anti-phishing tools for windows. Technical report, September 2006.

[17] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell. Client-side defense against web-based identity theft. In *Proceedings of the 11th Annual Network and Distributed System Security Symposium (NDSS'04)*, 2004.

[18] R. Dhamija and J. Tygar. The battle against phishing: Dynamic security skins. In *Proceedings of the 2005 symposium on Usable privacy and security (SOUPS'05)*, pages 77–88. ACM Press, 2005.

[19] S. Garera, N. Provos, M. Chew, and A. D. Rubin. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, pages 1–8, 2007.

[20] C. Ludl, S. McAllister, E. Kirda, and C. Kruegel. On the effectiveness of techniques to detect phishing sites. *Lecture Notes in Computer Science (LNCS)*, 4579:20–39, 2007.

[21] Y. Pan and X. Ding. Anomaly based web phishing page detection. In *Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC'06)*, pages 381–392, 2006.

[22] E. F. T. K. Sang and F. D. Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Computational Natural Language Learning (CoNLL-2003)*, pages 142–147, 2003.

[23] Y. Zhang, S. Egelman, L. Cranor, and J. Hong. Phinding phish: An evaluation of anti-phishing toolbars. In *Proceedings of the 14th Annual Network & Distributed System Security Symposium (NDSS 2007)*, 2007.

[24] Y. Zhang, J. Hong, and L. Cranor. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*, pages 639–648, 2007.