A Hierarchical Adaptive Probabilistic Approach for Zero Hour Phish Detection

Guang Xiang, Bryan A. Pendleton, Jason Hong, and Carolyn P. Rose

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA

Abstract. Phishing attacks are a significant threat to users of the Internet, causing tremendous economic loss every year. In combating phish, industry relies heavily on manual verification to achieve a low false positive rate, which, however, tends to be slow in responding to the huge volume of unique phishing URLs created by toolkits. Our goal here is to combine the best aspects of human verified blacklists and heuristic-based methods, i.e., the low false positive rate of the former and the broad and fast coverage of the latter. To this end, we present the design and evaluation of a hierarchical blacklist-enhanced phish detection framework. The key insight behind our detection algorithm is to leverage existing human-verified blacklists and apply the shingling technique, a popular near-duplicate detection algorithm used by search engines, to detect phish in a probabilistic fashion with very high accuracy. To achieve an extremely low false positive rate, we use a filtering module in our layered system, harnessing the power of search engines via information retrieval techniques to correct false positives. Comprehensive experiments over a diverse spectrum of data sources show that our method achieves 0% false positive rate (FP) with a true positive rate (TP) of 67.15% using search-oriented filtering, and 0.03% FP and 73.53% TP without the filtering module. With incremental model building capability via a sliding window mechanism, our approach is able to adapt quickly to new phishing variants, and is thus more responsive to the evolving attacks.

1 Introduction

Phishing is a social engineering attack, in which criminals build replicas of target websites and lure unsuspecting victims to disclose their sensitive information like passwords, personal identification numbers (PINs), etc. Exact numbers of direct damages done by phishing are hard to assess, in large part due to lack of data from organizations hit by phishing attacks. Estimates have ranged from a low of $61 million [15] to a high of $3.2 billion [1]. A significant proportion of those losses were caused by one particularly infamous group, known as the "rock phish gang", which uses phish toolkits to create a large number of unique phishing URLs, putting additional pressure on the timeliness and accuracy of blacklist-based anti-phishing techniques.

Generally, phish detection methods fall into two categories, i.e., those that perform URL matching via human verified blacklists and those that make use of heuristics via machine learning (ML) techniques. While the former has a very low false positive rate, human-verified blacklists do not generalize well to future unseen cases. For example, Sheng et al [21] showed that zero hour protection offered by major blacklist-based toolbars only has a true positive rate (TP) between 15% and 40%. Furthermore, human-verified blacklists can be slow to respond to new phishing attacks, and updating blacklists usually involves enormous human effort. For example, Phishtank [2] statistics in March 2009 show that it took on average 10 hours to verify that a URL was a phish. Finally, human-verified blacklists can be easily overwhelmed by automatically generated URLs. On the other hand, heuristic-based approaches enjoy the flexibility of being able to recognize new phish, but often lead to a relatively higher false positive rate. Concerns over liability for false positives have been a major barrier to deploying these technologies [20]. To underscore this point, Sheng et al [21] evaluated eight popular toolbars including Microsoft Internet Explorer, Firefox, Google Chrome, Norton 360, etc., all of which employ some level of human verification to achieve an extremely low FP in spite of the amount of human labor required, again primarily due to concerns over liability for false positives.

The goal of our work is to combine the best aspects of human verified blacklists and heuristics-based methods, and develop a reliable and robust method that is able to adaptively generalize to new attacks with reasonable TP while maintaining a close to zero FP. Our approach exploits the fact that a large number of current phishing attacks are created with toolkits, which tend to have a high similarity in terms of content. Our detection engine analyzes the content of phishing webpages on manually-verified URL blacklists via n-grams, and employs the shingling technique to identify near-duplicate phish in a probabilistic fashion. We also use a filtering module, which uses information retrieval (IR) techniques querying search engines to further scrutinize the legitimacy of a potential phish in an effort to control false positives. Our whole system is constantly updated by a sliding window upon the arrival of new phishing data, and is thus capable of adapting quickly to new phishing variants, while still maintaining a reasonable level of runtime performance. Under the optimal experimental setup, our method achieves a TP of 67.15% with 0% FP using search oriented filtering, and a TP of 73.53% and a FP of 0.03% without the filtering module, much better than blacklist-based methods in TP while comparable in FP. For applications like anti-phishing where FP is of paramount importance, a slightly lower TP is acceptable. Furthermore, we do not expect our approach to be used alone, but rather reside in the first part of a pipeline augmenting the existing system such as the commercial blacklists, thus fabricating a superior integrated solution.

We do not claim that our approach will solve the phishing problem. Rather, our specific claim is that we can augment existing blacklists in a very conservative manner using probabilistic techniques, with a very low FP, if not zero, and a reasonably good TP. Capable of identifying a fair amount of phishing attacks with no sacrifice on FP and considerably reducing the human effort involved

in manual verification, our approach significantly complements the prevalent blacklist-based methods, leveraging the manual labor that is already being used in verifying phishing sites. The major contributions of this paper are three fold.

1. We present the design of a novel hierarchical, content-based approach that leverages existing human-verified blacklists, by making use of shingling and information retrieval techniques to detect phish.
2. We demonstrate that with incremental model building via a sliding window mechanism, our approach is able to adapt quickly to the constantly evolving zero-hour phish attacks. Also, we only need the most recent 30 days' worth of data to achieve the same TP as using two months' worth of data, thus balancing accuracy with runtime efficiency.
3. By harnessing URL blacklists in a probabilistic fashion, we are able to leverage our approach to improve the coverage and timeliness of human-verified blacklists using considerably less human effort than existing techniques, without having to sacrifice the false positive rate. With only two weeks' worth of phish, our method achieves a TP of 65.02% with 0% FP using search oriented-filtering, and a TP of 71.23% and a FP of 0.03% without filtering.

2 Related Work

2.1 Methods for Automatic Phish Detection

A variety of techniques have been proposed for automatically detecting phishing web pages, and we will introduce some representative work in this section.

One camp exploits URL signatures to detect phish. Garera et al [13] identified a set of fine-grained heuristics from URLs, and combined them with other features to detect phish. Applying a logistic regression model on 18 features yielded an average TP of 95.8% and FP of 1.2% over a repository of 2508 URLs. Though interesting, this method has high variance in that URLs could be manipulated with little cost, causing the heuristics to fail.

Researchers have also devised a number of phish heuristics examining the content of web pages. Abu-Nimeh et al [8] adopted the bag-of-words strategy and used a list of words frequently found on phishing sites as features to detect phish, which is not expressive and easy to defeat by attackers. In [16], Ludl et al came up with a total of 18 properties solely based on the HTML and URL. The J48 decision tree algorithm was applied on these features and achieved a TP of 83.09% and a FP of 0.43% over a corpus with 4149 good pages and 680 phishing pages. However, heuristics purely based on DOM and URL are rather limited and may fail in capturing artfully designed phishing patterns. Zhang et al [23] proposed CANTINA, a content-based method using a linear classifier on top of eight features, achieving 89% TP and 1% FP on 100 phishing URLs and 100 legitimate URLs.

Another line of research focuses on discovering the intended phish brand to catch phish. Pan et al [18] proposed a method to extract the webpage identity from key parts of the HTML via the χ^2 test, and compiled a list of features based

upon the extracted identity. Trained with support vector machines (SVM), their features had an average FP of about 12%. 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. More recently, Xiang et al [22] proposed a hybrid detection model that recognizes phish by discovering the inconsistency between a webpage's true identity and its claimed identity via search engine and information extraction techniques. Their full integrated system achieved a TP of 90.06% and a FP of 1.95%.

Though the work in [23][22] also involve search engines, in this paper, we only resort to search engines in a postprocessing step to filter potential false positives in this paper while our core technique is the detection algorithm that exploits semantic similarity among the phishing attacks via the shingling technique. The only possible false positives generated in the detection phase are those well-known websites targeted by phishers, which guarantees the efficacy of our searching-based FP filtering method.

In addition to the past work above, anti-phishing toolbars are also available, many of which exploit human-verified blacklists to assure close-to-zero FP, such as NetCraft, Firefox 3, McAfee SiteAdvisor, etc.

Our goal in this paper is subtly different from the research above, in that we want to see how high our TP can be while maintaining close to 0% FP. As we noted in the introduction, industry has not adopted many of those heuristics above due to concerns about poor user experience for false positives as well as reasons of liability. Thus, our work here deliberately takes a conservative approach, though as we will show, we still get a reasonably good TP.

2.2 Toolkits for Creating Phishing Sites

In recent years, an increasingly large number of phishing webpages were automatically created by toolkits, which substantially increases the scale of attacks that criminals can attempt, while also countering current human-verified blacklists. For example, Cova et al [11] identified 584 phishing kits during a period of two months starting in April 2008, all of which were written in PHP. An analysis of rock-phishing sites by Moore et al [17] from February to April in 2007 reveals that 52.6% of all Phishtank reports were rock phish. One key observation of the rock phish is that their content is highly similar due to the way they are created, which is the property that our framework is based on. It is possible that criminals may modify their toolkits to include randomization to circumvent our detection mechanisms, and we discuss this issue towards the end of this paper.

3 A Multi-layered Phish Detection Algorithm

3.1 System Architecture

The overall architecture of our framework is shown in Fig.1. The first stage of processing involves filtering using domain whitelists, directly passing known

benign webpages. The detection engine employs a technique based on shingling to classify the remaining webpages, forwarding potential phish to the FP filter for further examination, which interacts with search engines to correct false positives. New phish from blacklists are added into our training set via a sliding window to update the detection model with the latest phishing patterns.

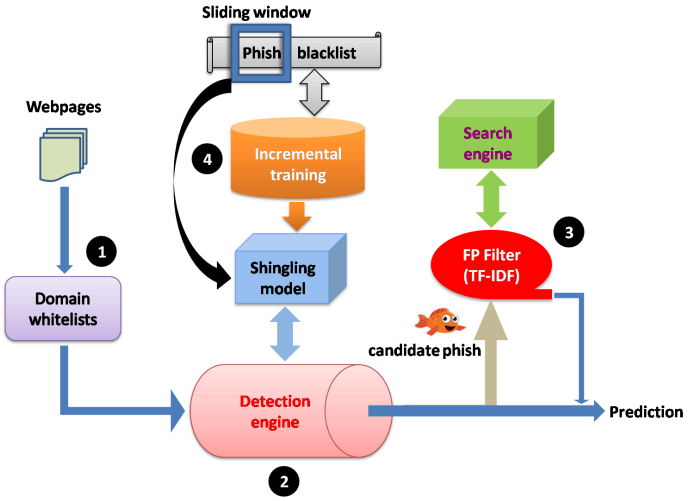


Fig. 1. System architecture. An incoming webpage is first checked against a small domain whitelist (1). If the page is not on the whitelist, our detection engine (2) compares the content of the webpage against the content of existing phish using the shingling algorithm. If a page is flagged as a potential phish, we check for false positives, resorting to search engines (3) if needed for additional verification. We use a sliding window (4) in the back-end to incrementally building the machine learning model as new phishing signatures arrive.

3.2 Shingling-based Probabilistic Matching

The essence of our detection algorithm is to do "soft" matching of a given web page against known phishing pages. The intuition here is that many phishing web pages are created by toolkits, and thus have many semantic similarities in terms of page content. Our detection method manipulates this semantic uniformity via soft matching, which allows more flexibility than the rigid URL matching adopted by major blacklist-based methods. Our early evaluations using exact matching with hash codes of page content turned out to be reasonably effective, but also brittle and easy to defeat. As such, we want to make our system robust to simple changes, thus raising the bar for criminals.

Shingling [10], a technique for identifying duplicate documents, examines the webpage content on a finer-grained level via the notion of n-gram, and measures

the inter-page similarity based on these basic units. N-grams are subsequences of n contiguous tokens. For example, suppose we have sample text ***connect with the eBay community***. This text has 3-grams {***connect with the, with the eBay, the eBay community***}. Shingling employs a metric named *resemblance* to calculate the percent of common n-grams between two webpages. More formally, let q and d represent a webpage being examined and a phishing page in the blacklist respectively. Let D represent the set of all training phish, and $S(p)$ denote the set of unique n-grams in p. The similarity metric *resemblance* $r(q, d)$ is then defined as $r(q, d) = |S(q) \cap S(d)|/|S(q) \cup S(d)|$. Our soft matching approach first generates the set of n-grams for each $d \in D$. We then compute $r(q, d)$ $\forall d \in D$ for a query page q, and fire an alarm whenever $r(q, d)$ exceeds a threshold t. We choose the optimal t via cross validation.

3.3 Search Engine Enhanced Filtering

As we will show later, shingling is effective in comparing a given web page against known phish. However, a potential problem is with false positives. More specifically, phishing web pages usually imitate legitimate web pages, which means that if there are no safeguards in place, shingling by itself is likely to label those target legitimate cases as phish as well. In addition to using domain whitelists to filter false positives, we propose a filtering algorithm leveraging the power of search engines via information retrieval techniques. This module, based on one heuristic in CANTINA [23], compensates for the incompleteness of domain whitelists, and is able to minimize FP even for less popular phishing target sites.

Our filtering module is triggered when the detection engine recognizes a candidate phish, and works by executing in Google queries composed of K top keywords chosen from the page content plus the webpage domain keyword [1] and examining the presence of the page domain in the top N search results. The final prediction is restored to "legitimate" if the top N entries subsume the page domain, and thus we no longer incorrectly label such sites as phish. The validity of this filtering algorithm is partially attributed to the fact that legitimate websites are very likely to be indexed by major search engines, while phishing sites are not, due to their short-lived nature and few in-coming links.

We currently use $K = 5$, $N = 30$ according to the tuning result in [23][22]. Candidate query terms on the page are ranked by the TF-IDF scoring function widely used in IR, which selects the terms that are most representative of the webpage content. The rationale is that search engines use TF-IDF when they match queries to documents in such a way that terms with high TF-IDF scores are the ones that have more influence over retrieval and ranking of documents.

3.4 Incremental Model Building via Sliding Window

To take the latest phishing signatures into our database and to improve the runtime performance of our whole system, we utilize a sliding window of the

[1] The domain keyword is the segment in the domain representing the brand name, which is usually the non-country code second-level domain or the third-level domain.

most recent phish from phish blacklists and incrementally build the detection model with those phishing web sites. In our evaluation, we show that discarding older data as the sliding window moves actually has little impact on accuracy.

Furthermore, a positive side effect of using a sliding window is that the time complexity of shingling is reduced from $O(|D|)$ to $O(|D_{win}|)$, where D_{win} represents all phishing data covered by the current sliding window win. Asymptotically, $|D_{win}|$ can be deemed as a large constant, and in light of this shrunk magnitude, we refrain from trading accuracy in exchange of speed via approximated algorithms as used in many applications [14]. For example, this sliding window could reduce a year's worth of phish to just a month's worth, achieving $\times 12$ runtime speedup without significantly sacrificing detection performance.

4 Experiment

4.1 Domain Whitelists

An enormous percentage of phishing frauds target well-known financial entities like eBay, Paypal, etc., by imitating their sites, and it is of practical value to pass those legitimate websites without feeding them to our detection engine. To reduce false positives and improve runtime performance, we quickly eliminate these known good sites through a whitelist. specifically, we collected known good domains from two sources. Google safe browsing provides a publicly-available database [3] with legitimate domains, and we obtained a total of 2758 unique domains from this whitelist after duplicate removal. Millersmiles [4] maintains an archive of the most common spam targets such as ebay, and we extracted 428 unique domains out of 732 entries after mapping organization names to domains and removing duplicates. In total, we had 3069 unique domains in our whitelist.

4.2 Webpage Corpus

Phishing sites are usually ephemeral, and most pages do not last more than a few days typically because they are taken down by the attackers themselves to avoid tracking, or taken down by legitimate authorities [21]. To study our approach over a larger corpus, we downloaded phishing pages when they were still alive and ran experiment offline. Our downloader employed Internet Explorer to render the webpages and execute Javascript, so that the DOM of the downloaded copy truly corresponds to the page content and thus gets around phishing obfuscations.

Our collection consists of phishing cases from PhishTank, and good webpages from seven sources. To eliminate the influence of language heterogeneity on our content-based methods, we only downloaded English webpages.

For phishing instances, we used the verified phishing URLs from the phish feed of Phishtank [5], a large community-based anti-phishing service with $38,324$ active accounts and $527,930$ verified phish [2] by the end of March 2010. We started downloading the feed in late February of 2009 and collected a total of 1175 phishing webpages from February 27, 2009 to April 2, 2009. All seven

legitimate corpus were downloaded after April 2, the details of which are given in Table 1. Note that the open directory project is the most comprehensive human-edited directory of the Web maintained by a vast community of volunteers, and by using this corpus, we want to verify that our algorithm achieves a very low FP on the low-profile and less popular sites.

Table 1. Legitimate collection with a total of 3336 web pages.

Source	Size	Crawling Method
Top 100 English sites from Alexa.com	958	Crawling homepages to a limited depth
Misc login pages	831	Using Google's "inurl" operator and searching for keywords like "signin"
3Sharp [19]	87	Downloading good webpages that still existed at the time of downloading
Generic bank category [6] on Yahoo directory	878	Crawling the bank homepages for a varying number of steps within the same domains
Other categories of Yahoo directory	330	Same as the generic bank category
The most common phishing targets	69	Saving login pages of those sites
The open directory project [7]	183	Downloading "least popular" pages with zero pagerank

4.3 Test Methodology

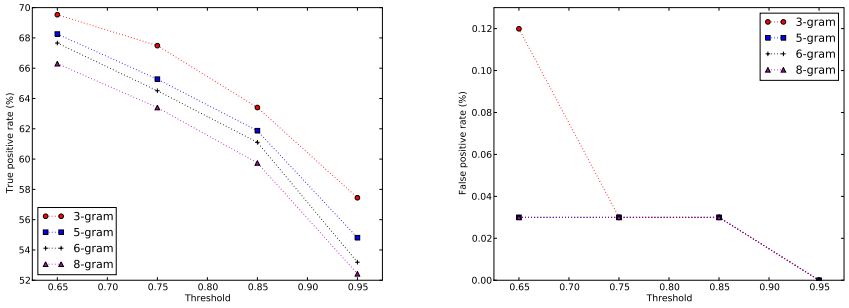
For notational convenience, we define in Table 2 the free variables in our context. Our experiment here focused on tuning these variables to optimize our results. To simulate a more realistic scenario, we processed data in chronological order in all of our experiments. In assessing TP, we move the sliding window of length L step by step along the time line and apply our detection algorithm to the webpages at each time point T_i using a shingling model built on the phishing data with time labels falling in window $[T_{i-L}, T_{i-1}]$. The FP is tested in a slightly different manner. In [12], Fetterly et al discovered through large-scale web crawling that webpage content was fairly stable over time, and based on that finding, we did not download the same set of legitimate pages at each time point but rather downloaded only once the whole set at a time later than all the phishing timestamps. Sliding windows of different sizes L are used similarly. Under all cases, four whitelist combinations are exercised with our detection algorithm, i.e., millersmiles, Google, none, and both whitelists.

4.4 Experimental Results

Shingling Parameter Tuning Figure 2 shows the validation performance under different values for n and t. For all n-grams in the evaluation, the TP monotonically decreased as we raised the resemblance bar higher. With a resemblance of 65%, shingling achieved over 66% TP under all shingle lengths, manifesting

Table 2. Definition of symbols.

Variable	Explanation	Variable	Explanation
G	granularity of time	L	sliding window length
W	whitelist	n	n-gram
r	resemblance	t	resemblance threshold



(a) TP under various r thresholds

(b) FP under various r thresholds

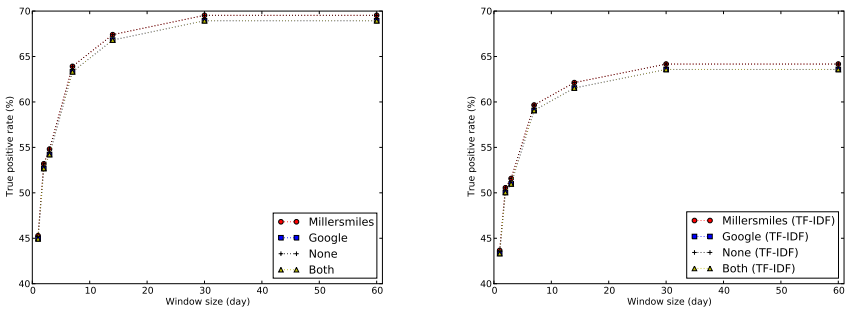
Fig. 2. Shingling parameter tuning ($L = 60$, $G =$ day, $W =$ millersmiles, no TF-IDF FP filtering). A tradeoff between TF and FP is an important factor in choosing t in the final model. As t is increased, the rate of detection drops and FP picks up. TP tops at 69.53% and FP reaches a culmination of 0.1199% under $n = 3, t = 0.65$. The other three FP curves $n = 5, 6, 8$ perfectly coincide.

the considerable similarity in content among phish due to rock phish. Although FP worsens as t and n decrease, we still stick to $n = 3, t = 0.65$ in the remaining evaluation of our experiment, hoping for the best TP performance and counting on the TF-IDF filtering algorithm to control false positives. The tuning results under all other configurations of G, L and W exhibit the same pattern, and we do not report them here.

Evaluation of True Positive Rate Figure 3 suggests that even with only one day's worth of training phish, our algorithm is able to detect around 45% phishing attacks, demonstrating the efficacy of our method and also proving the conjecture that mainstream phishing attacks are created by toolkits.

Another finding is that when using search engines to filter false positives (the right plot in Fig.3 and Fig.4), TP dropped as a side effect. An explanation is that some phishing URLs (2%/1% with a 1-day/1-hour sliding window) are actually returned among the top 30 entries when querying TF-IDF ranked terms plus the domain keyword on Google and are thus mistakenly filtered as legitimate.

Real-time application of our algorithm does not suffer from this false filtering problem as much as observed in our offline experiment. A semi-formal explana-



(a) TP without TF-IDF filter

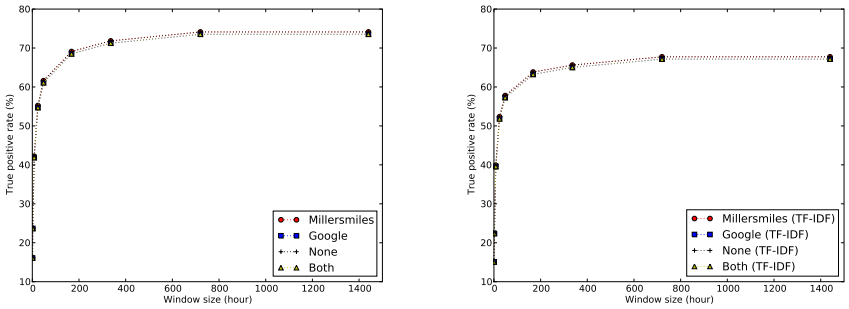
(b) TP with TF-IDF filter

Fig. 3. TP under various L (G = day) and W. Our approach achieves about 45% (no FP filtering) and 43% (with FP filtering) TP in all cases with only 1 day's worth of training phish, and around 69% (no FP filtering) and 64% TP (with FP filtering) with a 60-day window. FP filtering hurts TP, and a whitelist with only popular phishing targets beats a more comprehensive whitelist.

tion for this finding has two main points. First, when a new phish just comes out of attackers' workshop, few, if any, links point to that phishing site. As such, search engines are unlikely to return its domain as a top result; second, search engines might index the phish as time progresses when more links out in the web begin referring to it, however, the phish may have already become unavailable due to the short-lived nature of phishing activity and no harm will be done to the users even if it is incorrectly passed as a good page. The usefulness of this FP filtering module will become more evident when we embark on the analysis of FP in the following section.

Figures 3 and 4 suggest that the TPs under millersmiles whitelist are universally better than those under Google whitelist. Examining both whitelists reveals that millersmiles only contains a core group of the most spammed domains while the Google whitelist has many more less popular domains. None of the phishing domains in our corpus appear in the millersmiles whitelist, however, some do show up in the Google whitelist, among which is "free.fr", occurring 6 times in our phishing set. Those phish were thus erroneously filtered, lowering the TP inadvertently. This observation delivers a message about the use of domain whitelists, i.e., the quality of whitelists does impact TP and the optimal usage is to adopt a small core whitelist covering a group of popular spam target sites. Our detection method performed convincingly better with respect to TP when the model is iteratively built on a hourly basis.

Evaluation of False Positive Rate Figure 5 shows the FPs under different sliding window sizes and whitelists with no TF-IDF filtering. All four curves in both plots start with zero FPs, when L is minimum, and gradually escalate



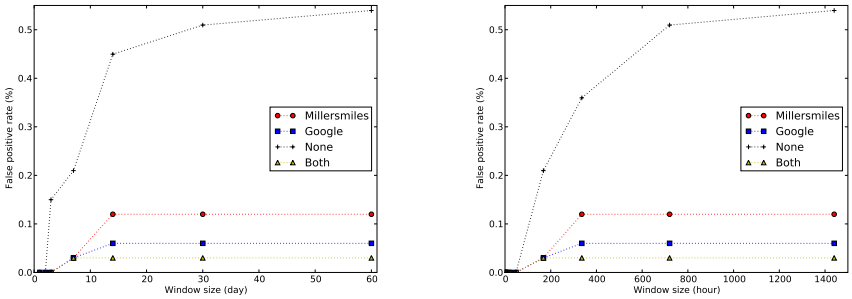
(a) TP without TF-IDF filter (b) TP with TF-IDF filter

Fig. 4. TP under various L (G = hour) and W. Under all whitelists, TP bottoms around 16% in all cases with a 1-hour window and peaks around 74% with a 1440-hour window without FP filtering; with FP filtering, TP bottoms around 15% with a 1-hour window and peaks around 67% with a 1440-hour window.

as more training phish are added to the model. Domain whitelists prove to be effective in suppressing false positives, with FPs of 0.1199%, 0.06%, 0.5396%, 0.03% for millersmiles, Google, none and both whitelists under both a 60-day window (left) and a 1440-hour window (right). With TF-IDF filtering, FPs are all zero under all circumstances, and we do not explicitly show the plots here.

Granularity of Time Unit for Window Size A comparison of TPs with day and hour based L (Table 4 in the appendix) shows that under sliding windows of identical time length, hour-level incremental model building outperformed day-level building, indicating the superior responsiveness of hourly updating. The largest gaps occurred at a window length of 1 day (24 hours), amounting to TPs of 9.95%, 9.78%, 9.95%, 9.78% with no FP filtering and 8.68%, 8.51%, 8.68%, 8.51% with FP filtering under four whitelist configurations. This disparity gradually diminished as L increased, which is reasonable in that as more and more phish are absorbed into the training set by the growing window, the tiny amount of shift in time relative to the window size no longer has as large of an impact as before. Surprisingly, simply with a 24-hour window, our algorithm was able to achieve over 50% TP under all whitelists and filtering setups.

As expected, the FPs under two time units in Table 5 in the appendix are identical except for one cell, since all legitimate pages in our web collection were downloaded after the phishing ones and regardless of time measurement (day or hour), the sliding window with the same length in terms of time actually covered roughly the same set of training phish. Interestingly, the FP filtering module successfully removed all the false positives, leading to zero FP under all experiment settings, at the cost of slight degradation on TP. Note that the evaluation of FP in our experiment is sound and thorough partially in that our legitimate



(a) FP vs window size (in number of days) (b) FP vs window size (in number of hours)

Fig. 5. FP under various L and W with no TF-IDF filtering. Under all whitelists, FP escalates with the growth of the sliding window size. FPs are zero when using TF-IDF to filter false positives under all settings and are not plotted here.

corpus contains a diverse variety of data including those categories that are the worst case scenarios for phish detection. As a result, the experimental result offers conservative statistics that are more meaningful to the actual adoption and deployment of our system. As suggested by the statistics in Table 4 and Table 5, another feature of our system is that it offers an adjustable range of performance depending on a user or provider's willingness to accept false positives.

Evaluation against Toolbars In [23], Zhang et al proposed CANTINA, a content-based method, which performed competitively 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 3 shows that our algorithm outperformed CANTINA significantly on FP, though its TP was inferior to CANTINA. Since the goal of our work is to achieve a high TP on the basis of maintaining a close-to-zero FP, we can accept this slight degradation on TP in exchange of a dramatic improvement on FP. All real-world anti-phishing applications call for an extremely low FP, and thus our solution is more effective and practical.

Table 3. Experiment reveals that our approach beats CANTINA significantly on FP with some degradation on TP, indicating that our method is more practical and effective in real-world scenarios. A combination of millersmiles and Google whitelists was used here.

	No FP filter	With FP filter	CANTINA
TP(%)	73.53	67.15	76.25
FP(%)	0.03	0.0	1.14

5 Discussion

In this section, we briefly discuss the merits and limitations of our current approach, and offer some ideas on how to address these problems.

5.1 Domain Whitelist and URL Blacklist

Domain whitelists have been shown in our experiment to be able to reduce FP. However, using whitelists has risks too. For example, phishing sites hosted on certain legal domains on our whitelists will be wrongly filtered as legitimate cases, as shown in Sect. 4.4. However, we can always reduce this impact by only using a core list of most targeted legal domains, whose defence systems are usually superb, and therefore, it will be extremely difficult for attackers to evade being detected by planting phishing sites into such legitimate domains.

Though our approach makes use of whitelists in the first step of the pipeline, it does not rely solely on them to achieve an acceptable FP. As suggested in Sect. 4.4, by only utilizing a small whitelist and mostly relying on the search engine-based filtering to slash false positives, our approach does not suffer from the incompleteness of whitelists and thus is scalable and realistic.

Currently, URL blacklists cannot detect new attacks unless the phishing URLs remain the same, which is unlikely due to the phishing nature of constantly avoiding tracking. Our work in this paper demonstrates a way to augment existing blacklists with conservative probabilistic strategies, and therefore we did not conduct an experiment specifically using traditional blacklists only.

5.2 Blacklist-based Soft Matching

Obtaining new phishing patterns in a timely fashion is critical to signature-based anti-phishing methods, and our approach addresses this problem by means of a sliding window that incrementally and constantly adds verified phish from blacklists into our database. Though the first few cases of new attacks are initially able to evade our detection, we only need to identify a few new phishing instances to update our model, subsequently being able to block the rest of the large volume of phishing attacks built by toolkits while maintaining a nearly zero FP.

This design philosophy emphasizes the adaptiveness and responsiveness of a usable phish detection system, and is a significant improvement over the traditional blacklist-based methods that are generally unable to cope with a high volume of unique phish URLs. To enlarge the range of phishing variants covered, our approach can be easily generalized to other phish feed like the APWG feed with the assistance of the whole anti-phishing community.

We could further exploit the dynamic aspects of phishing blacklists to improve our approach. For instance, we could prioritize our phish database by putting the phishing attacks with the most matches in a recent period of time in the top position for future comparison. In addition, although we currently focus on English websites, the general idea of our approach and the pattern of toolkit-based phishing attacks carry over to non-English sites, and our approach could be modified slightly to accommodate that change.

5.3 Effectiveness of TF-IDF Filtering

Our TF-IDF filter module has been shown to be effective by its extremely low number of FPs (see Sect. 4.4). Considering the fact that phishing activity always targets well-known brands due to its lucrative nature, false positives tend to be raised almost entirely on those popular target sites, which are very likely, if not almost certainly, to be indexed by major search engines. Accordingly, querying TF-IDF ranked terms plus the domain keyword will return the page domain as a top result with a high probability, thus successfully removing the false positive.

On the other hand, true phishing attacks are not as likely to be filtered by this module, thanks to the very nature that phishing sites rarely last long enough to be included in a web index. This will be difficult for phishers to change, because creating indexable, long-lived sites implies either access to stable hosting infrastructure, or that hacked sites will be able to go undiscovered for significant lengths of time. We believe the former is going to be too risky or expensive for phishers to engage widely in, and the latter is already being addressed by existing take-down approaches.

5.4 Legitimate Corpus

Our legitimate webpage collection mainly focuses on popular sites, commonly spammed sites, webpages with characteristics similar to phish (such as login forms), etc., and by appraising our idea on these hard cases, we actually provide worst case performance statistics, which is more beneficial for the real-life application and deployment that follow.

Our data set is by no means representative of what users experience during their everyday browsing. In [9], Bennouas et al proposed a random web crawl model and found through large-scale experiments that the in-degree, out-degree and pagerank of webpages follow power laws. Given the profit-driven nature of phishing activity, it is unlikely that the gigantic number of low-profile and less popular sites resemble the phishing pages with respect to the content, and not using those data in our experiment has no impact on the experiment result.

5.5 Runtime Performance

On a machine with 1.73 GHz CPU and 2.00G RAM running Windows XP, our detection algorithm took about 229.11 milliseconds on average to check each web page with a standard deviation of 220.99 milliseconds. Filtering via whitelists took roughly 0.18 milliseconds per URL. The phish detection phase via the shingling algorithm in our pipeline is essentially an parallel problem, and it should scale well because our phish database can be easily distributed into multiple computers and the process of matching page content probabilistically via database scanning can be easily parallelized.

We have two points of discussion here. First, we have not done many optimizations to our system for performance. Second, our approach is an embarrassingly parallel problem, one that scales well without a lot of effort, simply by

adding more computers. As such, existing blacklist services, ISPs, and takedown services could easily use our approach to improve their ability and timeliness in detecting phishing sites. The main limiting factor in terms of runtime performace is bandwidth. The TF-IDF filter in our system queries Google to verify the legitimacy of a webpage, which involves a round-trip traffic overhead on the web. However, this filter is only triggered when a page is sufficiently similar to an existing phishing signature, and considering the fact that the vast majority of the pages will not in any way resemble phishing attacks, the query frequency should be low enough. Moreover, caching search results on the client side is of paramount importance to speed up the performance, and may to a certain extent alleviate the network traffic problem.

5.6 How Phishers may Respond

We do not claim that our approach will solve the phishing problem. Rather, our specific claim is that we can augment existing blacklists in a very conservative manner using probabilistic techniques, with good results in terms of TPs and FPs. Taking a wider perspective, criminals will inevitably come up with countermeasures to the specifics of any deployed technique. What we offer here is a new way of thinking about how to effectively combine human-verification with ML and IR techniques, to increase the effectiveness of existing blacklists.

Phishers could try to penetrate our method by HTML obfuscation tricks such as injecting garbage text to the DOM with tiny or invisible fonts, background color, 100% transparency, multiple i-frames. These are, however, more of an implementation issue than a design one, and we can easily peel off those special effects and extract intentionally separated text by manipulating the DOM in the preprocessing step. Our system also cannot detect Flash-based phishing, and would require other techniques to detect these. The use of other types of popular obfuscation techniques such as doorway pages, chains of redirections, URL cloaking and so on is also futile in front of our algorithm. The reason is that no matter how many irrelevant intermediate steps attackers try to involve, web users will land in the actual phishing webpage eventually, and our content-based detection idea still applies. As a matter of fact, such tricks in hope of obfuscating online customers and anti-phishing algorithms turn out to be beneficial to our method in that search engines are even less likely to crawl those phishing sites given such gimmicks, and our search-oriented filtering module is thus more unlikely to incorrectly filter the corresponding phishing attacks as legitimate cases. These kinds of tricks could also be new features for machine learning algorithms as well, again since few legitimate sites would use such techniques.

Another likely countermeasure that criminals would attempt is to have toolkits include some element of randomization, making it harder to do soft matching. This, however, is not hard to cope with. If the randomization is in the invisible part of the DOM, our argument in the beginning of the previous paragraph still applies, and we can easily extract the real content. Should the random elements be added to the web page content, we could tune the resemblance threshold t accordingly and still achieve a reasonable detection rate. On the other hand, there

is a limit on how much random noise attackers could add before web users start feeling suspicious about the legitimacy of the web pages. Furthermore, restricting people's awareness while hindering probabilistic matching simultaneously by adding noise is not an easy task, which would make the process of designing phish toolkits very difficult and thus significantly limits the vast production of phishing attacks, rendering the cost-benefit undesirable for the criminals.

It is very hard for attackers to elevate the FP of our approach, since the design of legitimate webpages and the crawling process of search engines are beyond their control. It is even harder to trick search engines to give their phishing sites higher rankings, due to the scrutiny of search engines, short-lived nature of phishing behavior and negligible popularity scores of the phishing sites.

6 Conclusion

In this paper, we presented a system that combined human-verified blacklists with information retrieval and machine learning techniques, yielding a probabilistic phish detection framework that can quickly adapt to new attacks with reasonably good true positive rates and close to zero false positive rates.

Our system exploits the high similarity among phishing web pages, a result of the wide use of toolkits by criminals. We applied shingling, a well-known technique used by search engines for web page duplication detection, to label a given web page as being similar (or dissimilar) from known phish taken from blacklists. To minimize false positives, we used two whitelists of legitimate domains, as well as a filtering module which uses the well-known TF-IDF algorithm and search engine queries, to further examine the legitimacy of potential phish.

We conducted extensive experiments using phish from Phishtank and legitimate web pages from seven different sources. These experiments showed that our proposed method had a TP of 73.53% and a FP of 0.03% without TF-IDF filtering, and a TP of 67.15% and zero FP with TF-IDF filtering under the optimal setting. Moreover, our approach is able to adapt quickly to zero-hour attacks by incrementally building the model via a sliding window with a few new phishing instances out of a huge magnitude of phishing attacks created by toolkits, thus providing a feasible framework for industry to vastly improve the existing limited blacklists without increasing their false positives. This sliding window mechanism also leads to good balance between accuracy and runtime efficiency: with only two weeks' worth of training phish, our method had a TP of 65.02% with 0% FP using search oriented-filtering, and a TP of 71.23% and a FP of 0.03% without FP filtering.

Acknowledgements This work has been supported by NSF grants CCF-0524189 and DGE-0903659. Additional support has been provided ARO research grant DAAD19-02-1-0389 to Carnegie Mellon University's CyLab, and the CMU/Portugal Information and Communication Technologies Institute.

References

1. http://www.gartner.com/it/page.jsp?id=565125
2. http://www.phishtank.com/stats.php
3. http://sb.google.com/safebrowsing/update?version=goog-white-domain:1:1
4. http://www.millersmiles.co.uk/scams.php
5. http://data.phishtank.com/data/online-valid/
6. http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Banking/Banks/
7. http://rdf.dmoz.org/
8. Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S.: A comparison of machine learning techniques for phishing detection. In: Proceedings of the anti-phishing working groups (APWG) 2nd annual eCrime researchers summit. pp. 60–69 (2007)
9. Bennouas, T., de Montgolfier, F.: Random web crawls. In: Proceedings of the 16th international conference on World Wide Web (WWW'07). pp. 451–460 (2007)
10. Broder, A.Z., Glassman, S.C., Manasse, M.S., Zweig, G.: Syntactic clustering of the web. In: Proceedings of the sixth international conference on World Wide Web. pp. 1157–1166 (1997)
11. Cova, M., Kruegel, C., Vigna, G.: There is no free phish: An analysis of 'free' and live phishing kits. In: Proceedings of the 2nd USENIX Workshop on Offensive Technologies (WOOT'08) (2008)
12. Fetterly, D., Manasse, M., Najork, M.: On the evolution of clusters of near-duplicate web pages. In: Proceedings of the First Conference on Latin American Web Congress. pp. 37–45 (2003)
13. Garera, S., Provos, N., Chew, M., Rubin, A.D.: A framework for detection and measurement of phishing attacks. In: Proceedings of the 2007 ACM Workshop on Recurring Malcode. pp. 1–8 (2007)
14. Henzinger, M.: Combinatorial algorithms for web search engines: three success stories. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. pp. 1022–1026 (2007)
15. Herley, C., Florncio, D.: A profitless endeavor: phishing as tragedy of the commons. In: Proceedings of the 2008 workshop on New security paradigms. pp. 59–70 (2009)
16. Ludl, C., McAllister, S., Kirda, E., Kruegel, C.: On the effectiveness of techniques to detect phishing sites. Lecture Notes in Computer Science (LNCS) 4579, 20–39 (2007)
17. Moore, T., Clayton, R.: Examining the impact of website take-down on phishing. In: Proceedings of the Anti-phishing Working Groups (APWG) 2nd annual eCrime Researchers Summit. pp. 1–13 (2007)
18. Pan, Y., Ding, X.: Anomaly based web phishing page detection. In: Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC'06). pp. 381–392 (2006)
19. 3sharp report: Gone phishing: Evaluating anti-phishing tools for windows. Tech. rep. (September 2006), http://www.3sharp.com/projects/antiphishing/gone-phishing.pdf
20. Sheng, S., Kumaraguru, P., Acquisti, A., Cranor, L., Hong, J.: Improving phishing countermeasures: An analysis of expert interviews. In: Proceedings of the 4th APWG eCrime Researchers Summit (2009)
21. Sheng, S., Wardman, B., Warner, G., Cranor, L., Hong, J., Zhang, C.: An empirical analysis of phishing blacklists. In: Proceedings of the 6th Conference on Email and Anti-Spam (2009)

22. Xiang, G., Hong, J.: A hybrid phish detection approach by identity discovery and keywords retrieval. In: Proceedings of the 18th International Conference on World Wide Web (WWW'09). pp. 571–580 (2009)
23. Zhang, Y., Hong, J., Cranor, L.: Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th International Conference on World Wide Web (WWW'07). pp. 639–648 (2007)

Appendix: Detailed Results under Various Settings

Table 4. TP (%) under day/hour-measured sliding window. Under all settings, shingling with hour-level incremental model building is more responsive to phishing attacks, attaining higher TPs under all L values. Our approach achieved almost optimal TP with only 1 month's worth of training phish.

	No TF-IDF filtering											
	Window size (day)						Window size (hour)					
Whitelist	1	2	7	14	30	60	24	48	168	336	720	1440
Millersmiles	45.28	53.19	63.91	67.4	**69.53**	**69.53**	55.23	61.62	69.11	71.83	**74.13**	**74.13**
Google	44.94	52.68	63.32	66.81	**68.94**	**68.94**	54.72	61.11	68.51	71.23	**73.53**	**73.53**
None	45.28	53.19	63.91	67.4	**69.53**	**69.53**	55.23	61.62	69.11	71.83	**74.13**	**74.13**
Both	44.94	52.68	63.32	66.81	**68.94**	**68.94**	54.72	61.11	68.51	71.23	**73.53**	**73.53**
	With TF-IDF filtering											
	Window size (day)						Window size (hour)					
Whitelist	1	2	7	14	30	60	24	48	168	336	720	1440
Millersmiles	43.66	50.55	59.66	62.13	**64.17**	**64.17**	52.34	57.79	63.83	65.62	**67.74**	**67.74**
Google	43.32	50.04	59.06	61.53	**63.57**	**63.57**	51.83	57.28	63.23	65.02	**67.15**	**67.15**
None	43.66	50.55	59.66	62.13	**64.17**	**64.17**	52.34	57.79	63.83	65.62	**67.74**	**67.74**
Both	43.32	50.04	59.06	61.53	**63.57**	**63.57**	51.83	57.28	63.23	65.02	**67.15**	**67.15**

Table 5. FP (%) under day/hour-measured sliding window. Whitelists lessen the FPs, reaching 0.1199%, 0.06%, 0.5396%, 0.03% respectively with the millersmiles, Google, none and both whitelists at $L = 60$ days or $L = 1440$ hours. The search engine oriented filtering step significantly significantly improves the FPs , downsizing FP values in all settings to zero.

	No TF-IDF filtering											
	Window size (day)						Window size (hour)					
Whitelist	1	2	7	14	30	60	24	48	168	336	720	1440
Millersmiles	0.00	0.00	0.03	0.1199	0.1199	0.1199	0.00	0.00	0.03	0.1199	0.1199	0.1199
Google	0.00	0.00	0.03	0.06	0.06	0.06	0.00	0.00	0.03	0.06	0.06	0.06
None	0.00	0.00	0.2098	0.4496	0.5096	0.5396	0.00	0.00	0.2098	0.3597	0.5096	0.5396
Both	0.00	0.00	0.03	0.03	0.03	0.03	0.00	0.00	0.03	0.03	0.03	0.03
	With TF-IDF filtering											
	Window size (day)						Window size (hour)					
Whitelist	1	2	7	14	30	60	24	48	168	336	720	1440
Millersmiles	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Google	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
None	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Both	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00