Marchal, Samuel; Saari, Kalle; Singh, Nidhi; Asokan, N

Know Your Phish: Novel Techniques for Detecting Phishing Sites and Their Targets

Published in:
IEEE 36th International Conference on Distributed Computing Systems (ICDCS)

DOI:
10.1109/ICDCS.2016.10

Published: 10/08/2016

Document Version
Peer reviewed version

Please cite the original version:
Marchal, S., Saari, K., Singh, N., & Asokan, N. (2016). Know Your Phish: Novel Techniques for Detecting Phishing Sites and Their Targets. In *IEEE 36th International Conference on Distributed Computing Systems (ICDCS)* (pp. 323 - 333). (International Conference on Distributed Computing Systems. Proceedings). IEEE. https://doi.org/10.1109/ICDCS.2016.10

Know Your Phish: Novel Techniques for Detecting Phishing Sites and their Targets

Samuel Marchal*, Kalle Saari*, Nidhi Singh† and N. Asokan*‡
*Aalto University
†Intel Security
‡University of Helsinki
Email: {samuel.marchal,kalle.saari}@aalto.fi; nidhi.singh@intel.com; asokan@acm.org

***Abstract*—Phishing is a major problem on the Web. Despite the significant attention it has received over the years, there has been no definitive solution. While the state-of-the-art solutions have reasonably good performance, they require a large amount of training data and are not adept at detecting phishing attacks against new targets.**

In this paper, we begin with two core observations: (a) although phishers try to make a phishing webpage look similar to its target, they do not have unlimited freedom in structuring the phishing webpage; and (b) a webpage can be characterized by a small set of key terms; how these key terms are used in different parts of a webpage is different in the case of legitimate and phishing webpages. Based on these observations, we develop a phishing detection system with several notable properties: it requires very little training data, scales well to much larger test data, is language-independent, fast, resilient to adaptive attacks and implemented entirely on client-side. In addition, we developed a target identification component that can identify the target website that a phishing webpage is attempting to mimic. The target detection component is faster than previously reported systems and can help minimize false positives in our phishing detection system.

I. Introduction

Phishing webpages ("phishs") lure unsuspecting web surfers into revealing their credentials. As a major security concern on the web, phishing has attracted the attention of many researchers and practitioners. There is a wealth of literature, tools and techniques for helping web surfers to detect and avoid phishing webpages. Nevertheless, phishing detection remains an arms race with no definitive solution. State-of-the-art large scale real-time phishing detection techniques [**?**] are capable of identifying phishing webpages with high accuracy (>99%) while achieving very low rates of misclassifying legitimate webpages (<0.1%). However, many of these techniques, which use machine learning, rely on millions of static features, primarily taking the bag-of-words approach. This implies two major weaknesses: (a) they need a huge amount of labeled data to train their classification models; and (b) they are language- and brand-dependent and not very effective at identifying new phishing webpages targeting brands that were not already observed in previous attacks. Commercial providers of phishing detection solutions struggle with obtaining and maintaining labeled training data. From the deployability perspective, solutions that require minimal training data are thus very attractive.

In this paper, we introduce a new approach that avoids these drawbacks. Our goal is to identify whether a given webpage is a phish, and, if it is, identify the *target* it is trying to mimic. Our approach is based on two core conjectures:

- **Modeling phisher limitations**: To increase their chances of success, phishers try to make their phish mimic its target closely and obscure any signal that might tip off the victim. However, in crafting the structure of the phishing webpage, phishers are restricted in two significant ways. First, external hyperlinks in the phishing webpage, especially those pointing to the target, are to domains *outside the control* of phishers. Second, while phishers can freely change most parts of the phishing page, the latter part of its domain name is *constrained* as they are limited to domains that the phishers control. We conjecture that by modeling these limitations in our phishing detection classifier, we can improve its effectiveness.
- **Measuring consistency in term usage**: A webpage can be represented by a collection of key terms that occur in multiple parts of the page such as its body text, title, domain name, other parts of the URL etc. We conjecture that the way in which these terms are used in different parts of the page will be different in legitimate and phishing webpages.

Based on these conjectures, we develop and evaluate a phishing detection system. We use comparatively few (212) but relevant features. This allows our system, even with very little labeled training data, to have high accuracy and low rate of mislabeling legitimate websites. By modeling inherent phisher limitations in our feature set, the system is resilient to adaptive attackers who dynamically change a phish to circumvent detection. Our basic phishing detector component (Section IVPhishing Detection Systemsection.4) does not require online access to centralized information and is fast. Therefore, it is highly suited for a privacy-friendly client-side implementation. Our target brand identification component (Section **??**) uses a simple technique to extract a set of *keyterms* characterizing a webpage and, in case it is a phish, uses the keyterms set to identify its target. Both components eschew the bag-of-words approach and are thus not limited to specific languages or targeted brands.

We claim the following contributions:

- a new set of features to detect phishing webpages (Section IV-BComputing Featuressubsection.4.2) and a classifier, built using these features, with the following properties that distinguish it from previous work:
  - it learns a generalized model of phishing and legitimate webpages from a small training set (few thousands).
  - it is language- and brand-independent.
  - its features are extracted only from information retrieved by a web browser from the webpage and it does not require online access to centralized information. Hence it admits a client-side-only implementation that offers several advantages including (a) better privacy, (b) real-time protection and (c) resilient to phishing webpages that return different contents to different clients.
- comprehensive evaluation of this system, showing that its accuracy (>99%) and misclassification rate (<0.1%) are comparable to prior work while using significantly smaller training data. (Section V-CPhishing Webpage Classificationsubsection.5.3)

II. Background

A. Phishing

Phishing refers to the class of attacks where a victim is lured to a fake webpage masquerading as a target website and is deceived into disclosing personal data or credentials. Phishing campaigns are typically conducted using spam emails to drive users to fake websites [**?**]. Impersonation techniques range from technical subterfuges (email spoofing, DNS spoofing, etc.) to social engineering. The former is used by technically skilled phishers while unskilled phishers resort to the latter [**?**]. Phishing webpages mimic the look and feel of their target websites [**?**]. In order to make the phishing webpages believable, phishers may embed some content (HTML code, images, etc.) taken directly from the target website and use relatively little content that they themselves host [**?**]. This includes outgoing links pointing to the target website. They also use keywords referring to the target in different elements of the phishing webpage (title, text, images, links) [**?**], [**?**], [**?**], [**?**]. In this paper, our focus is on detection of phishing webpages created by an attacker and hosted on his own web server or on someone else's compromised web server.

B. URL Structure

Webpages are addressed by a uniform resource locator (URL). Fig. 1Structure of a URLfigure.caption.1 shows relevant parts in the structure of a typical URL. It begins with the *protocol* used to access the page. The fully qualified domain name (*FQDN)* identifies the server hosting the webpage. It consists of a registered domain name (*RDN*) and prefix which we refer to as *subdomains*. A phisher has full control over the *subdomains* portion and can set it to any value. The *RDN* portion is constrained since it has to be registered with a domain name registrar. *RDN* itself consists of two parts: a *public suffix* (*ps*) preceded by a *main level domain* (*mld*). The URL may also have a *path* and *query* components which, too, can be changed by the phisher at will. We use the term *FreeURL* to refer to those parts of the URL that are fully controllable by the phisher.

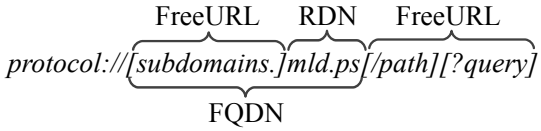

Fig. 1: Structure of a URL

Consider an example URL:
https://www.amazon.co.uk/ap/signin?_encoding=UTF8
We can identify the following components:

- *protocol = https*
- *FQDN = www.amazon.co.uk*
- *RDN = amazon.co.uk*
- *mld = amazon*
- *FreeURL* = {*www*, */ap/signin?_encoding=UTF8*}

C. Data Sources

From analyzing phishing webpages, we identify the following data sources, available to a web browser when it loads a webpage, that can be useful in detecting phishing webpages:

- *Starting URL*: the URL given to the user to access the website. It can be distributed in emails, instant messages, websites, documents, etc.
- *Landing URL*: the final URL pointing to the actual content presented to the user in his web browser. This is the URL present in the browser address bar when the page is completely loaded.
- *Redirection chain*: the set of URLs crossed to go from the starting URL to the landing URL (including both).
- *Logged links*: the set of URLs logged by the browser while loading the page. They point to sources from which embedded content (code, images, etc.) in the webpage are loaded.
- *HTML*: the HTML source code of the webpage and IFrames included in the page. We consider four elements extracted from this source code:
  - *Text*: text contained between <*body*> HTML tags (actually rendered on user's display).
  - *Title*: text contained between <*title*> HTML tags (appears in the browser tab title).
  - *HREF links*: the set of URLs representing outgoing links in the webpage.
  - *Copyright*: the copyright notice, if any, in Text.
- *Screenshot*: an image capture of the loaded webpage.

III. Design Overview

A. Modeling Phisher Limitations

In Section II-BURL Structuresubsection.2.2, we saw that even on systems they control, phishers are constrained from

freely constructing URLs to pages they host. Similarly, in Section II-APhishingsubsection.2.1, we saw that in order to maximize the believability of their phishing sites, phishers include content from URLs outside their control. Thus, we divide the data sources from Section II-CData Sourcessubsection.2.3 into subcategories according to the level of *control* phishers may have on them and the *constraints* on phishers.

Control: URLs from *logged links* and *HREF links* are subdivided into *internal* and *external* according to their *RDN*. The set of *RDN*s extracted from URLs involved in the redirection chain are assumed to be under the control of the webpage owner. Any URLs that include these *RDN*s are marked *internal*. Other *RDN*s are assumed to be possibly outside the control of the webpage owner. URLs containing such *RDN*s are marked *external*.

Constraints: Within a URL, we distinguish between *RDN*, which cannot be freely defined by the webpage owner, and (*FreeURL*), which can be.

B. Extracting Term Distributions

The primary technique of a phisher is essentially social engineering: fooling a victim into believing that the phishing webpage is the target [**?**]. Thus, it is plausible that lexical analysis of the data sources will help in identifying phishing webpages: we conjecture that legitimate webpages and phishing webpages differ in the way terms are used in *different locations* in those pages. To incorporate measurements of such term usage consistency, we first define what "terms" are and how they are extracted from a webpage. Let A be the set of the 26 lowercase English letters: $A = \{a, b, c, ..., x, y, z\}$. We extract terms from a data source as follows:

- canonicalize letter characters by mapping upper case characters, accented characters and special characters to a matching letter in A; e.g., $\{$ *B*, *β*, $\hat{b}$, $\hat{b}$ $\} \rightarrow b$.
- split the input into substrings whenever a character $c \notin A$ is encountered.
- throw away any substring whose length is less than 3.

Let $T = A^n | n \geq 3$ be the set of all possible terms. Suppose $T_S = \{t_{i \in \{1;m\}} \in T\}$ was extracted from a data source S and t_i occurs with probability p_i. The set of m pairs $(t_i, p_i) \in T \times]0, 1], i \in \{1; m\}$ represents the *term distribution* D_S of *S*.

TABLE I: Term distributions

Distribution	Data source
D_{text}	Text
D_{title}	Title
$D_{copyright}$	Copyright notice
D_{image}	Webpage screenshot
D_{start}	Starting URL – *FreeURL*
D_{land}	Landing URL – *FreeURL*
D_{intlog}	Internal logged links – *FreeURL*
$D_{intlink}$	Internal HREF links – *FreeURL*
$D_{startrdn}$	Starting URL – *RDN*
$D_{landrdn}$	Landing URL – *RDN*
D_{intrdn}	Internal links (HREF and logged) – *RDN*
D_{extrdn}	External logged links – *RDN*
D_{extlog}	External logged links – *FreeURL*
$D_{extlink}$	External HREF links – *FreeURL*

Table ITerm distributionstable.caption.2 defines the term distributions we consider. The external sources $extrdn$, $extlog$, $extlink$ are those assumed to be outside the control of the webpage owner. *RDN* data sources $startrdn$, $landrdn$, $intrdn$ are constrained by DNS registration. The rest is controlled by the webpage owner without constraints. The $image$ data source is composed of terms extracted by optical character recognition (OCR) from the screenshot of a rendered webpage.

C. Architecture

Our overall design consists of a phishing webpage detection system (Section IVPhishing Detection Systemsection.4) and a target identification system (Section **??**). The phishing detection system is a classifier that identifies phishing webpages based on a set of newly introduced features. The target identification system identifies if a given webpage is a phish by finding its target. Both systems can be used in a pipeline: the phishing detection system tentatively identifies a potential phish, which can be fed to the target identification system to infer the purported target.

IV. Phishing Detection System

A. Feature Set Requirements

We consider some facts of phishing detection in order to deduce requirements that a feature set must have:

Generalizability: Accumulating ground truth phishing and legitimate data is challenging. Phishing websites have very short lifetimes [**?**] and can display different content depending on a browser's user-agent or user's geographic location. Labeled phishing and legitimate resources are often defined by URLs (*e.g.* PhishTank[1]). Assigning correct labels (phish, non-phish) to these URLs is difficult. But even if the initial labeling was done correctly, information on the pages pointed by them can also evolve over time: a legitimate domain name can be hijacked to host phishing content for a while or a phishing domain name can be parked or changed to contain empty content after a short uptime. Therefore, crawling a set of labeled URLs to gather ground truth data often leads to noisy datasets that further require manual checking and cleaning up. Thus it is desirable to select a feature set that allows a model to be learned from *as small a training set as possible* while remaining applicable to far larger test datasets. Using a much larger test set than the training set also allows the detection and avoidance of overfitting [**?**].

Adaptability: Several automated classification techniques [**?**], [**?**], [**?**] rely on a static set of features learned from a training set such as the bag-of-words model or "term frequency-inverse document frequency" (TF-IDF) [**?**] computation. Such feature models are language-dependent and vary with training sets. Using such features shows [**?**] that certain terms such as *paypal* are dominant features. Thus the efficacy of such models on phishs that masquerade as previously unknown targets or brands is questionable. In addition, phishers can adaptively

[1] PhishTank (https://www.phishtank.com/)

modify the content of their phish to circumvent detection by such static models, e.g., by using words that typically occur in legitimate webpages. An adaptable feature set must be *independent of learning instances*, preferably defined manually with motivated reasons, and be resilient to adaptive attacks.

Usability: It is desirable that features are computable on an end user system *without relying on online access to centralized servers* or proprietary data (*e.g.* Google PageRank). This preserves user privacy since the scheme does not require users to disclose their browsing history to an outside entity.

Computational Efficiency: Features must be *quickly computable* to allow integration with real time detection systems that do not impact users' web surfing experience.

B. Computing Features

We now introduce 212 features and motivate their selection. We intend to capture the constraints and degree of control discussed earlier (Section III-AModeling Phisher Limitationssubsection.3.1) as well as consistency checking of term usage (Section III-BExtracting Term Distributionssubsection.3.2). We group features into five categories (Table IIFeature setstable.caption.3).

TABLE II: Feature sets

Name	Count	Type
f_1	106	URL
f_2	66	Term usage consistency
f_3	22	Usage of starting and landing *mld*
f_4	13	*RDN* usage
f_5	5	Webpage content
f_{all}	212	Entire feature set

URL: First we define nine statistical features related to the lexical composition of URLs (Table IIIURL featurestable.caption.4). Feature 2 is meant to identify strings in *path* and *query* that look like domain names. Phishing URL and domain name obfuscation techniques [**?**] tend to produce long URLs composed of many terms. This is the rationale for features 3-8. The popularity rank of the domain (feature 9) is based on a fixed, previously downloaded list of the Alexa top million domain names[2]. If a domain is not in this list, feature 9 takes the default value of 1,000,001.

All nine features are extracted from the starting URL (9) and landing URL (9). The mean, median and standard deviation values are computed for features 3-9 on the following sets of URLs: internal logged links, external logged links, internal HREF links and external HREF links ($4 * 7 * 3$). Feature 1 is computed on these sets as a ratio of URLs using *https* over the total count of URLs for each set ($4 * 1$). Feature 2 is computed only for the starting and landing URLs. Thus, the complete URL-based feature set (f_1) consists of 106 features: $9 + 9 + 4 * (7 * 3 + 1) = 106$.

Term usage consistency: The second set of features (f_2) captures the consistency of term usage between different types (controlled vs. uncontrolled; constrained vs. unconstrained) of data sources in the page. Using 12 term distributions (we discard $D_{copyright}$ and D_{image}) defined in Section III-BExtracting Term Distributionssubsection.3.2 we define 66 features ($12 * 11/2$) depicting the similarity of pairs of sources by computing pairwise Hellinger Distance between their distributions. The Hellinger Distance [**?**] is a metric used to quantify the dissimilarity between two probabilistic distributions P and Q. It is an instance of f-divergence that is symmetric and bounded in $[0, 1]$. The value 1 represents complete dissimilarity ($P \cap Q = \emptyset$) and the value 0 means that P and Q are the same probabilistic distribution.

TABLE III: URL features

#	Description
1	protocol used (http/https)
2	count of dots '.' in *FreeURL*
3	count of level domains
4	length of the URL
5	length of the *FQDN*
6	length of the *mld*
7	count of terms in the URL
8	count of terms in the *mld*
9	Alexa ranking of the *RDN*

Usage of starting and landing *mld*: Legitimate websites are likely to register a domain name reflecting the brand or the service they represent. However, phishers often use domain names having no relation with their target [**?**]. Hence, we expect the starting *mld* and/or the landing *mld* to appear in several sources extracted from a legitimate webpage while phishing webpages should not have this characteristic. We define 22 features (f_3) inferring the usage of the starting and landing *mld* in the text, the title and *FreeURL* of the logged links and HREF links. 12 binary features are set to 1 if the starting/landing *mld* appear in D_{text}, D_{title}, D_{intlog}, D_{extlog}, $D_{intlink}$ or $D_{extlink}$ (6*2); 10 features are the sum of probability from terms of D_{title}, D_{intlog}, D_{extlog}, $D_{intlink}$ and $D_{extlink}$ that are substrings of starting/landing *mld* (5*2). D_{text} is not considered since it is often composed of many short irrelevant terms that match several parts of a *mld*.

***RDN* usage**: We define 13 features (f_4) related to *RDN* usage consistency. We compute statistics related to the use of similar and different *RDN*s in starting URL, landing URL, redirection chain, loaded content (logged links) and HREF links. We expect legitimate webpages to use more *internal RDN*s and less redirection than phishing webpages [**?**].

Webpage content: Finally, five features (f_5) count the number of terms in the text and the title (2), and the number of input fields, images and IFrames (3) in the page. Phishing pages tend to have minimal text to circumvent text-based detection techniques [**?**] and use more images and HTML content loaded from other sources. In addition, since phishing attacks seek to steal user data, phishing webpages often contain several input fields [**?**].

It is worth noting that while we use terms to compute our feature set, it is not based on any observed language or term usage knowledge. The computation relies solely on the

[2]Alexa (http://www.alexa.com/)

information gathered through a web browser albeit we use a local copy of Alexa ranking list. Hence, it makes the feature set adaptable and usable as well as fast to compute once the data sources are available. Since the feature set is small (212) we expect it to have good generalizability.

C. Phishing Detection Model

To use our feature set for discriminating phishing from legitimate webpages, we use a supervised machine learning approach. In supervised machine learning, a classification model is learned from observations over a set of data labeled with several classes. The learned model is used to predict the class of unlabeled instances. We select Gradient Boosting [**?**] to build the classification model. It was selected because [**?**] (a) of its strong ability to select and weight the most relevant features and (b) boosting algorithms are known to be fairly robust to overfitting, enabling the resulting model to have good generalization capabilities.

Gradient Boosting predicts the class of an unknown instance by computing values defined in $[0, 1]$ that gives the confidence of the instance to belong to a given class. In the case of predicting only two classes, the confidence value v_1 for one class is equal to $1 - v_2$, where v_2 is the confidence value for the other class. A *discrimination threshold* predicts, according to the computed confidence values, the class of an instance. By tuning this threshold, we can favor the prediction of one class over the other. The variation of the discrimination threshold over $[0, 1]$ is used to evaluate the accuracy of a given model by examining how false positive rate varies with true positive rate (ROC) or precision varies with recall.

V. Evaluation

In this section, we present the performance evaluation of the phishing detection system and the target identification method presented in Sections IVPhishing Detection Systemsection.4 and **??** respectively.

A. Experimental Setup

Our system is composed of five Python modules:

Webpage scraper is only required for experiments to gather the information sources defined in Section II-CData Sourcessubsection.2.3. It can also be used for offline analysis. The scraper is implemented as a monitored Firefox web browser (Selenium[3]) that extracts the data sources while visiting a webpage at a given URL. It saves the data in json format and a screenshot of the webpage.

Feature extractor extracts the 212 features (Section IV-BComputing Featuressubsection.4.2) from the data sources in the webpage and builds a feature vector.

Classifier takes the feature vector and a previously learned classification model as input to predict the class, phishing or legitimate, of a webpage. The implementation of the Gradient Boosting is provided by the Scikit Learn[4] Python package.

Keyterms extractor infers the keyterms of a webpage using data gathered by the scraper.

Target identifier predicts the likelihood of a webpage being a phish. In case of a phish, the modules also identifies its target.

B. Evaluation Datasets

We obtained URLs from two sources in order to gather ground truth data of phishing and legitimate webpages (Table IVDatasets descriptiontable.caption.5). Neither dataset contains personal data. We will make both datasets available for research use.

The phishing URL sets (**Phish**) were obtained through the community website PhishTank. We conducted three different collection "campaigns". The first resulted in *phishTrain* which was used for training the phishing detection classifier. The second, collected at a later point in time, resulted in *phishTest* which was used as the test set. The last, *phishBrand*, was used for evaluating our target identification scheme (Section V-DTarget Identificationsubsection.5.4). It consists of 600 phishing webpages for each of which we manually identified the target, resulting in a total of 126 different targets. Each campaign consisted of checking for new entries in PhishTank every hour and scraping the webpages for those URLs. These are in several languages. The datasets were further manually cleaned to remove any legitimate or unavailable websites and parked domain names. Table IVDatasets descriptiontable.caption.5 provides a detailed description of these datasets including the date and the count of elements before and after cleaning.

TABLE IV: Datasets description

Set	Name	Date (2015)	Initial	Clean
Phish	*phishTrain*	Jul-23/Aug-3	1213	1036
	phishTest	Sep-13/Sep-24	1553	1216
	phishBrand	Sep-22/Sep-28	600	600
Leg	*legTrain*	Jul-15/Jul-22	5000	4531
	English	Aug-17/Sep-23	100,000	–
	French	Sep-28	10,000	–
	German	Sep-29	10,000	–
	Italian	Sep-30	10,000	–
	Portuguese	Oct-1	10,000	–
	Spanish	Oct-2	10,000	–

The legitimate URLs (**Leg**) were provided by Intel Security[5]. We processed them same way as for the phishing URLs. Intel gave us several datasets. First, an English training set (*legTrain*) of 5,000 legitimate webpages was cleaned up to remove unavailable websites and dead links. Six larger test sets of webpages in different languages (English, French, German, Portuguese, Italian and Spanish) were gathered and did not receive any cleaning treatment. A detailed description of these sets is provided in Table IVDatasets descriptiontable.caption.5 as well. The variety and popularity of the URLs in the test set is reflected in the fact that 65,302 (43.5%) of the 150,000 test URLs in **Leg** have *RDN*s ranked in Alexa top 1M.

[3]Selenium HQ (http://www.seleniumhq.org/)

[4]Scikit Learn (http://scikit-learn.org/)

[5]Intel Security (http://www.intelsecurity.com/)

C. Phishing Webpage Classification

We now present detailed evaluation of our phishing detection method. We focus on three primary aspects of classification performance. First is *accuracy* which entails precision, recall and false positive rate. Second is *Receiver Operating Characteristic (ROC)*, which shows the change of false positive rate with respect to true positive rate. Third is *scalability* where we evaluate how accuracy changes as we scale from small to large test datasets. We evaluate the performance of our method across six different languages so as to demonstrate its language independence. The evaluation scenario for all languages consists of the same learning stage on *legTrain* and *phishTrain* (5,567 webpages), being the oldest captured datasets. Prediction is based on *phishTest* and each individual language-specific test dataset of legitimate URLs.

TABLE V: Detailed accuracy evaluation for six languages

Language	Pre.	Recall	F_1-score	FP Rate	AUC
English	0.956	0.958	0.957	0.0005	0.999
French	0.970	0.958	0.964	0.0036	0.997
German	0.981	0.958	0.970	0.0022	0.998
Portuguese	0.967	0.958	0.962	0.004	0.997
Italian	0.982	0.958	0.970	0.0021	0.998
Spanish	0.982	0.958	0.970	0.0021	0.998

Accuracy: The detailed evaluation results for precision, recall and false positive rate, using legitimate datasets of six different languages are shown in Table VDetailed accuracy evaluation for six languagestable.caption.6. These values were obtained by setting the discrimination threshold of Gradient Boosting to 0.7, which favors the prediction of legitmate webpages ($[0, 0.7[$) over phishs ($[0.7, 1]$). In this table, we see that our method achieves significantly high precision for all languages (0.95–0.98). This holds for recall as well (around 0.95). Hence, the F_1-score, which is the harmonic mean of precision and recall, is also significantly high (0.95–0.97). The false positive rate is significantly low, i.e., in the range of 0.0005–0.004, across all languages.

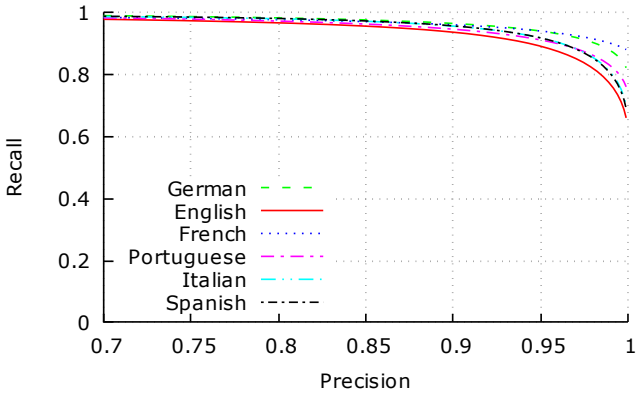

Fig. 2: Precision vs recall evaluation

In many large-scale, real-world scenarios (especially in web security domain), a machine learning model is considered

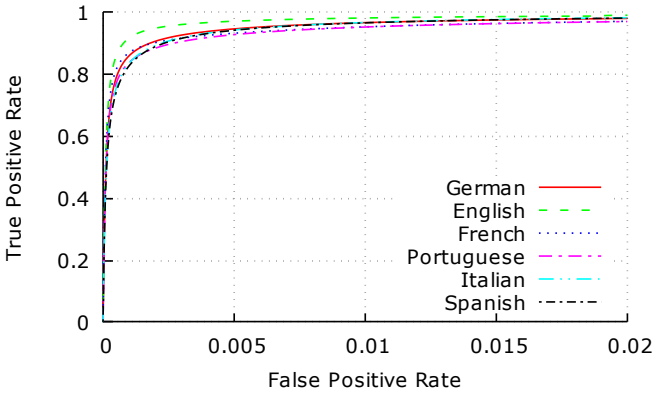

Fig. 3: ROC evaluation results for six languages

usable only if it achieves high precision (e.g., 0.9 or 0.95) with significant recall (e.g., 0.5 or 0.6) [**?**]. In order to test our method against this criterion, we evaluated how recall of the proposed method changes with precision by varying the discrimination threshold from 0 to 1. The result is shown in Fig. 2Precision vs recall evaluationfigure.caption.7 where we see that when the precision is higher than 0.9, the recall for all languages is significantly high and is always in the range of 0.64–0.98. This shows that from the accuracy perspective, our method is readily applicable in large-scale, multi-lingual business scenarios.

ROC: Another metric for predictive performance of the proposed method is ROC and corresponding AUC (Area Under the Curve). Along the lines of accuracy evaluation, we examine the ROC and AUC metrics across all languages. The objective of ROC evaluation is to examine the increase in false positive rate with the increase in true positive rate while varying the discrimination threshold of the classifier. The evaluation results for all languages are shown in Fig. 3ROC evaluation results for six languagesfigure.caption.8. We see that, at the significantly high true positive rate of 0.9, the false positive rate for all languages is less than 0.008 which is considered quite low. As the true positive rate increases to around 0.95, the false positive rate does not increase much. Even at true positive rate of 0.98, the false positive rate stays substantially low at 0.02. In line with these results, the AUC is around 0.999 for all languages, as shown in Table VDetailed accuracy evaluation for six languagestable.caption.6. Note that these results are consistent across all languages, which is very desirable in a multi-lingual phishing detection scenario.

Scalability: We now examine the effect of scale on the predictive performance of our method, i.e., if the size of the test dataset increases considerably with time, then what effect does it have on the precision, recall and false positive rate? We initialize our test set with 10,000 legitimate and 100 phishing examples extracted randomly from the English dataset and *phishTest* respectively. Thereafter, we iteratively increment the size of the test set by 10,000 legitimate webpages and 100 phishs randomly picked from the remaining instances of the

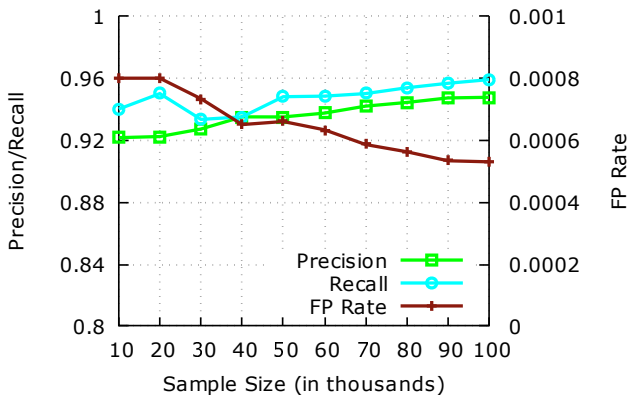

Fig. 4: Performance vs the scale of data

English dataset and *phishTest*.

The results are shown in Fig. 4Performance vs the scale of datafigure.caption.9, where we see that precision as well as recall increase with scale, whereas the false positive rate decreases. This indicates that the increase in the number of errors (i.e., false positives and false negatives) is significantly less than the increase in the size of test set, which causes the increase in overall precision and recall and decrease in the false positive rate as we scale up the size of test set. This kind of predictive performance on large test set, while learning a model from a small training set, is exactly what is required in a desirable machine learning model that is deemed fit for usage in large-scale practical scenarios.

	Median	Average	StDev
Webpage scraping	12787	12798	4898
Loading data	1	2	2
Features extraction	890	1282	1586
Classification	< 1	< 1	< 1
Total (no scraping)	891	1283	1588

TABLE VI: Processing time (milliseconds)

Table VIProcessing time (milliseconds)table.caption.10 depicts the median, average and standard deviation of the time taken by each operation involved in phishing webpage classification on a laptop with 2.7GHz Intel Core i5 processor and 16GB memory. We can see that most of the time is dedicated to webpage scraping, which is not part of the classification process in the case of client-side implementation. Except that, the total median classification time is 891 milliseconds, showing that the system is able to render a decision in less than 1 second. These performance figures are based on a standalone Python prototype. Subsequently, we also implemented an optimized JavaScript version as a browser add-on which exhibits better performance characteristics [**?**].

D. Target Identification

To assess the performance of target identification, we used the 600 phishing webpages of *phishBrand*. Since target identification can provide up to three candidate targets for an analyzed webpage, we evaluate the likelihood of the correct target being part of top-1, top-2 and top-3 results provided by our scheme. Table VIITarget identification resultstable.caption.11 presents the count of correctly identified targets, unknown targets and missed targets considering these three sets. The last column gives the success rate of each method. The 17 pages with unknown target corresponds to webpages including only some input fields and no hint about the target. We were not able to infer the target with manual analysis. These webpages with unknown targets are thus included in the computing of the success rate. The accuracy of identifying the correct target (top-1) is 90.5%. If the criteria is identifying the correct target among a possible set of 3 (top-3) then the accuracy increases to 97.3%. These results are comparable to the best state of the art method for target identification [**?**] that reaches a 92.1% success rate.

TABLE VII: Target identification results

Targets	Identified	Unknown	Missed	Success rate
top-1	526	17	57	90.5%
top-2	558	17	25	95.8%
top-3	567	17	16	97.3%

To see how the target identification system can complement our phishing detection system we fed the former with misclassified legitimate webpages identified in Section V-CPhishing Webpage Classificationsubsection.5.3 when assessing phishing detection with the English dataset. 53 out of 100,000 legitimate webpages were misclassified. The target identification system identified four of these as phish with an identified target. 10 were considered as suspicious (no target identified and no legitimate confirmation) and 39 were confirmed as legitimate.

Considering these results, using the target identification in a second step for instances identified as phishs by the phishing detection system can be beneficial. On the *English* dataset, it would reduce the false positive rate to 0.0001, which is equal to the best state-of-the-art phishing detection system [**?**]. However, according to accuracy in target identification (97.3%) it would as well reduce the number of identified phishs while keeping precision and recall over 0.90.

VI. Discussion

A. Relevance of Feature Sets

We have seen in Section V-CPhishing Webpage Classificationsubsection.5.3 that our features set yielded results that outperform previous work. The main reason for this improvement is the new separation scheme applied to data sources related to their level of control and constraints (Section III-AModeling Phisher Limitationssubsection.3.1). Analysing the weight of each features in the learnt classification model we observed that individually, features belonging to f_2, were the most relevant. We omit details about feature analysis for lack of space, see our research report [**?**] for details. This explains the reason why we obtained comparable results to the best existing techniques [**?**], while relying on less features and training data.

In addition, we assessed that our feature set meets the requirements introduced in Section IV-AFeature Set

Requirementssubsection.4.1. It has good generalizability being able to learn a classification model from few thousand instances and accurately predicting the class of 100,000+ unknown webpages. It is adaptable and language/brand independent achieving comparable performances across different languages. It is usable as it does not rely on online access to centralized information and is fast to render a decision with a median processing time lower than 1 second.

B. Limitations

The main strength of our technique, its language independence, is though its main weakness. We chose to split strings according to any characters that are not part of the English dictionary and to only consider terms composed of at least three characters to discard stop words and recurrent short terms having no meaning. This raised some issues in term distribution comparisons. Long subdomains such as *theinstantexchange* or *insuranceservicenow* were considered as single term. In contrast, short domain name string corresponding to brand and composed of separating characters (digit, hyphen, etc.) such as *dl4a*, *s2mr* or *e-go* were split and the resulting terms were discarded as too short. The inconsistent usage of abbreviations or acronyms like *intl* for *international* or *pfa* for *premier financial online* also had a negative impact. Similarity of synonyms cannot be inferred. Most misclassified legitimate webpages (>50%) had one of these characteristics. Despite these misclassifications we achieve a low false positive rate (0.0005). Many of these misclassified instances can be identified as legitimate by the target identification system.

A second limitation relates to the identification of some empty/unavailable webpages and parked domain names as phishs. The former is explained by the lack of information contained in empty/unavailable webpages. Several parked domain names are domains that have been used for malicious purposes like phishing [**?**] and are thus obfuscated *FQDN*s registered to trap users. Moreover, parked domain names use similar composition schemes and obfuscation techniques as phishing domains [**?**] such as typosquatting [**?**]. Parked domain names and phishing domain names have other common characteristics. Parked domains are involved in advertisement networks [**?**] and the delivered ad content tends to be correlated with the domain name parked, for instance ads for Amazon Inc. are delivered for the *RDN amaaon.com*. From the point of view of our classification system, these parked pages have the same characteristics as phishing pages. This misclassification of unavailable and parked domain names is not of major concern since, for the former no content access is prohibited by the system since the link point empty resources. For the latter, domain parking is considered as spam by major Internet actors (*e.g.* Google) and some efficient state-of-the-art techniques [**?**] or the target identification system can be applied to discard these webpages from phishing identification.

A last limitation was the low accuracy observed in classification of IP-based phishing URLs. Out of 25 such URLs in *phishTest*, only 19 were correctly classified rendering a lower recall (0.76) than the global recall presented by the system (>0.95). The reason is that *FQDN*s based term distributions for such URLs are empty leading to several null features. However, such URLs represent less than 2% (41) of the URLs present in all phishing datasets and is thus not a major limitation.

Although we did not observe this in our datasets, webpages whose content is in one alphabet and URL in another may be misclassified. So far we have only tested webpages in European languages. Classifier performance on pages in other languages may be lower.

C. Evasion Techniques

As we saw, one way to evade detection is to use IP-based URLs. These are less likely to be detected by our system. However, relying on IP address rather than domain names deprives phishers from the flexibility brought by the DNS to change the hosting location of their phishing content while keeping the same link. Moreover, IP blacklisting is widely used to prevent access to malicious hosting infrastructure, so phishers would have to face other issues.

Another evasion technique is to limit the text content available in a webpage: use few external links, do not load external content and build short URLs [**?**]. We observed some of these techniques actually being used individually in webpages of both phishing datasets used for evaluation. They did not impact classifier performance because even though they prevent some features from being computed, others, such as those based on title, starting/landing URL and logged links could still lead to effective detection of phishs. Simultaneous use of multiple evasion techniques may impact classifier performance. However, using such subterfuges would impact the quality of the phishing webpage and reduce the number of victims.

A final probable evasion technique is to use typosquatting domains and misspelled terms in the different data sources we analyze. When different but similar terms like *paypal*, *paypaI* or *paipal* are used in different sources, our distributions comparison metric would not infer any similarity. The classifier would thus probably conclude that the webpage is legitimate. However, the presence of references to the target would disclose the real target. In addition, misspellings may tip-off potential victims.

For target identification, the best evasion technique is not to provide any indication about the target in the webpage and rather focus on using lures in the message containing the link to the fake website. But this has two negative effects, first, the phishing webpage seems less legitimate and second, the phisher exposes himself to alternative target identification techniques applied to other content than webpages [**?**].

VII. Related Work

The obfuscation and mimicry characteristics of phishing webpages have been the basis of several solutions proposed for phishing detection and target identification.

Phishing webpage detection: Analysis of the content [**?**], [**?**] and code execution (*e.g.* the use of javascript, pop-up

Technique	Testing set Legitimate	Phish	Legitimate set	Train /Test	Leg /Phish	Evaluation	FPR	Pre.	Recall	Acc.
Cantina [?]	2,100	19	English	-	110/1	no learning	0.03	0.212	0.89	0.969
Cantina+ [?]	1,868	940	several	1/4	2/1	old/new	0.013	0.964	0.955	0.97
Xiang *et al.* [?]	7,906	3,543	several	-	2/1	no learning	0.019	0.957	0.9	0.955
Ma *et al.* [?]	15,000	20,500	DMOZ	1/1	3/4	cross-valid	0.001	0.998	0.924	0.955
Whittaker *et al.* [?]	1,499,109	16,967	several	6/1	90/1	old/new	0.0001	0.989	0.915	0.999
Thomas *et al.* [?]	500,000	500,000	several	4/1	1/1	cross-valid	0.003	0.961	0.734	0.866
Ramesh *et al.* [?]	1,200	3,374	top Alexa	-	1/3	no learning	0.005	0.998	0.996	0.996
Chen *et al.* [?]	404	1,945	top Alexa	9/1	1/5	cross-valid	0.007	0.992	1	0.994
Our method	100,000	1,216	English	1/18	85/1	old/new	0.0005	0.956	0.958	0.999
Our method	150,000	1,216	several	1/27	125/1	old/new	0.001	0.857	0.958	0.998

TABLE VIII: Phishing detection system performances comparison

windows, etc.) [?] of a webpage provides relevant information to identify phishing webpages. Some detection methods rely on URL lexical obfuscation characteristics [?], [?] and webpage hosting related features [?], [?] to render a decision about the legitimacy of a webpage. The visual similarity of a phishing webpage with its target was also exploited to detect phishs [?], [?]. Phishing detection based on visual similarity presuppose that a potential target is known a priori. In contrast, our approach is to discover the target.

Multi-criteria methods [?], [?] have been proved the most efficient to detect phishing websites. These techniques use a combination of webpage features (HTML terms, links, frame, etc.), connection features (HTML header, redirection, etc.) and host based features (DNS, IP, ASN, geolocation, etc.) to infer webpage legitimacy. They are implemented as offline systems checking content pointed by URLs to automatically build blacklists. This process induces a delay of several hours [?] that is problematic in the context of phishing detection, since phishing attacks have a median lifetime of a few hours [?]. In addition, it is reportedly costly [?] and use [?] some proprietary features preventing usage on the end-user devices. The identification method uses machine learning techniques fed with hundreds of thousands of features. These features are mostly static and learned from training sets containing data such as IP address, Autonomous System Number (ASN), bag-of-words for different data sources (webpage, URL, etc.). This limits the generalizability of the approach as it requires large training datasets, numbering hundreds of thousand of webpages [?].

Other methods focused, as we do, on the study of terms that compose the data sources of a webpage [?], [?]. Cantina [?], [?] was among the first systems to propose a lexical analysis of terms that compose a webpage. In Cantina [?] key terms are selected using TF-IDF to provide a unique signature of a webpage. Using this signature in a search engine, Cantina infers the legitimacy of a webpage. A similar method [?], based on TF-IDF and Google search, checks for inconsistency between a webpage identity and the identity it impersonates to identify phish. The main difference between these methods and ours is language independence since these methods rely on TF-IDF computation to infer their keyterms.

Table VIIIPhishing detection system performances comparisontable.caption.12 presents comparative performances results of our phishing detection system to the most relevant state-of-the-art systems. It presents the size of the testing sets used to evaluate each system and the provenance of the legitimate set, showing how representative the set is. For example, using popular websites (such as top Alexa sites) [?], [?] as the legitimate set is not representative. The ratio of training to testing instances indicates the scalability of the method and the ratio of legitimate to phishing instances shows the extent to which the experiments represents a real world distribution ($\approx 100/1$) [?], [?]. We also identify the evaluation method (e.g., cross validation vs. training with old data and testing with new data). Finally, we present several metrics for assessing the classification performance. If data for any of the columns were missing from the original paper describing the system, we estimated them. For comparison purposes, if several experimental setups were proposed in a paper, we selected the most relevant to assess their practical efficacy using the following ordered criteria:

1) learning and testing instances are different,
2) the ratio of legitimate to phishing in the testing set is representative of real world observations ($\approx 100/1$),
3) the learning set is older than the testing set,
4) the false positive rate (FPR) is minimized.

We can see that among the eight most relevant state-of-the-art techniques, only two [?], [?] have comparable false positive rates to ours (≤ 0.001). A low false positive rate is paramount for a phishing detection technique, since this relates to the proportion of legitimate webpages to which a user will be incorrectly denied. The technique proposed by Ma *et al.* [?] has a lower accuracy than in our system ($0.955 < 0.999$). In addition, they use a testing set that does not represent real world distribution (3 legs/4 phishs) and use a cross-validation that does not assess scalability of the approach with a 1/1 ratio for learning to testing instances. Whittaker *et al.* [?] report results similar to us in several metrics. However, they use a *huge training set (>9M instances)* and their test set is actually *smaller* than the training set (a sixth, at 1.5M)! Scalability and language/brand independence are likely to be poor since they use 100,000 mostly static features (bag-of-words).

In contrast to the state-of-the-art in phishing detection, our solution is language independent, scalable, requires much smaller training sets than test sets, and does not rely on real-

time access to external sources, while performing better than or as well as the state-of-the-art.

Target identification: One proposal [**?**] was to use a similar technique as Cantina with keywords retrieval and Google search to discover a list of potential target as the top results of the search, but the authors do not report accuracy figures for target identification. HREF links have been used to build community graphs of webpages. By counting the mutual links between two webpages and further performing visual similarity analysis between suspicious webpages, Liu *et al.* [**?**] identify the target of a given phishing website with an accuracy of 92.1%. However, this technique is slow because of the need to crawl many additional websites to build the community graph. Conditional Random Fields and Latent Dirichlet Allocation (LDA) [**?**] have been applied to phishing email content to identify their target [**?**] with a success rate of 88.1%.

The technique we propose, in contrast to previous techniques is language independent for keyterms inference. It is as efficient as any state-of-the-art solutions achieving a maximum success rate of 90.5-97.3%.

VIII. Conclusion

We presented novel techniques for efficiently and economically identifying phishing webpages and their targets. By using a set of features that capture inherent limitations that phishers face, our system has excellent performance and scalability while requiring much smaller amounts of training data. We have also implemented a fully client-side phishing prevention browser add-on implementing this technique [**?**].

IX. Acknowledgments

This work was supported in part by the Academy of Finland (grant 274951) and Intel Collaborative Research Center for Secure Computing (ICRI-SC). We thank Craig Olinsky, Alex Ott and Edward Dixon for valuable discussions.