Everything Is in the Name – A URL Based Approach for Phishing Detection

Harshal Tupsamudre(✉), Ajeet Kumar Singh, and Sachin Lodha

TCS Research, Pune, India
{harshal.tupsamudre,ajeetk.singh1,sachin.lodha}@tcs.com

Abstract. Phishing attack, in which a user is tricked into revealing sensitive information on a spoofed website, is one of the most common threat to cybersecurity. Most modern web browsers counter phishing attacks using a blacklist of confirmed phishing URLs. However, one major disadvantage of the blacklist method is that it is ineffective against newly generated phishes. Machine learning based techniques that rely on features extracted from URL (*e.g.*, URL length and bag-of-words) or web page (*e.g.*, TF-IDF and form fields) are considered to be more effective in identifying new phishing attacks. The main benefit of using URL based features over page based features is that the machine learning model can classify new URLs on-the-fly even before the page is loaded by the web browser, thus avoiding other potential dangers such as drive-by download attacks and cryptojacking attacks.

In this work, we focus on improving the performance of URL based detection techniques. We show that, although a classifier trained on traditional bag-of-words features (tokenized using special characters) works well in many cases, it fails to recognize a very prevalent class of phishing URLs that combines a popular brand with one or more words (e.g., `www.paypalloginsecure.com` and `paypalhelpservice.simdif.com`) among others. To overcome these flaws, we explore various alternative feature extraction techniques based on word segmentation and $n-$grams. We also construct and use a phishy-list of popular words that are highly indicative of phishing attacks. We verify the efficacy of each of these feature sets by training a logistic regression classifier on a large dataset consisting of 100,000 URLs. Our experimental results reveal that features based on word segmentation, phishy-list and numerical features (*e.g.*, URL length) perform better than all other features, as measured by misclassification and false negative rates.

Keywords: Phishing detection · Machine learning · Social engineering attacks

1 Introduction

Phishing is a form of social engineering attack that exploits the weakest link in the security chain, *i.e.*, humans. The attack typically starts with an email

S. Dolev et al. (Eds.): CSCML 2019, LNCS 11527, pp. 231–248, 2019.
https://doi.org/10.1007/978-3-030-20951-3_21

campaign that appears to come from a legitimate entity such as PayPal. The email lures the recipient into clicking a URL, which leads the user to a website designed to look legitimate but is not. When the user enters sensitive data such as passwords or credit card numbers, the fraudulent website records the information and sends it back to the attacker. Phishing attacks are extremely successful. According to 2018 Verizon's data breach investigation report [26], phishing is the third most common threat vector for data breaches and 4% of users click on any given phishing campaign. Phishing attacks are not only increasing in number, but they are also getting more sophisticated every day. The Anti Phishing Work Group (APWG) identified a total of 151,014 unique phishing websites in the third quarter of 2018. About half of these websites (49.4%) were hosted on infrastructure with HTTPS and SSL certificates, whereas at the end of 2016, the number of phishing websites using HTTPS were merely less than 5% [1].

The security community has invested a great deal of effort in developing detection countermeasures against phishing attacks. Most phishing detection techniques can be broadly classified into three categories, blacklist based, heuristics based and machine learning based [14]. Currently, Google Safe Browsing [3] is the most popular blacklisting service and is used by several web browsers including Chrome, Firefox and Safari to prevent users from visiting phishing websites. Microsoft offers similar such service known as SmartScreen and is used in the Internet explorer. The blacklist method is easy to implement, however one major disadvantage of this method is that it lacks the ability to protect against zero-hour phishing attacks. According to one study [25], 63% of the phishing campaigns end within the first two hours, whereas 47% to 83% of phishing URLs appeared in blacklists only after 12 h.

The heuristics based approaches exploit common characteristics found in the previously reported phishing attacks in order to detect new attacks. Few examples of heuristic tests are as follows:

- if the host-name portion of a URL is an IP address, the URL is phishing.
- if an organization's name (*e.g.*, PayPal) is present in a URL path but not in the primary domain, the URL is phishing.
- if hyphen is present in a primary domain, the URL is phishing.
- if password field is present in a web page, the website is phishing.

However, the use of heuristics can be tricky as it requires choosing the right weights for each heuristic check, and if not done properly it runs the risk of misclassifying legitimate websites. Machine learning algorithms, on the other hand, automatically determines best weights for all features (heuristic checks) using a database of training examples. In the machine learning approach, the problem of phishing detection is formulated as a binary classification task with two classes: *phishing* (positive class) and *valid* (negative class). The features required for training the classifier are mainly extracted from the URL [16,17,23] or web page [7,31] or both [14,19]. While the use of web page features may lead to better classification accuracy, the main benefit of using URL based features is that the resulting model can classify new URLs on-the-fly even before the page is loaded by the web browser, thus avoiding other potential dangers like drive-by

download and cryptojacking attacks. Further, page based detection techniques suffer from performance issues, as many of these [7,31] work only after the entire web page is rendered, and there is a possibility that users may have divulged sensitive data before the page is detected as phishing.

The term URL is an abbreviation of Uniform Resource Locator, the global address of documents and other resources on the World Wide Web. A URL has three main components: (i) protocol, (ii) hostname, and (iii) path. The hostname specifies the server on which the resource is located and the path specifies the location of the document on the server. The hostname is further divided into two sub-parts: subdomain and domain. The path is also divided into three sub-parts: directory, file name and arguments. An example is shown in Fig. 1. In the figure, the term TLD stands for top-level domain.

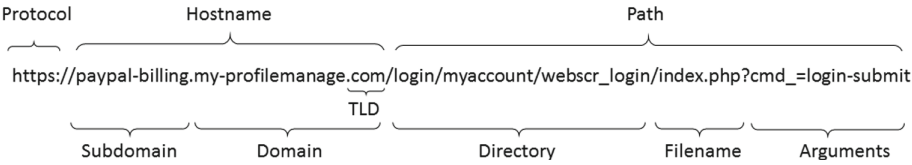

Fig. 1. Different components of a URL

URL features are of two types: *lexical* features and *external* features [16, 17]. Lexical features are those which can be quickly extracted from the URL string such as the length of the URL, the number of dots in the URL and the bag-of-words features. External features, on the other hand, require queries to remote servers (*e.g.*, whois lookup and DNS resolution) which introduces additional overhead and consume more resources at the client, *e.g.*, battery life and bandwidth of devices. Researchers [16] showed that the performance of a classifier that uses only lexical features is comparable to the one that uses full features (lexical + external). Therefore, lexical features are more appropriate for implementing an anti-phishing solution at the client side.

1.1 Contributions

Although a classifier trained on conventional lexical features [16,17] performs well in many cases, in this paper, we demonstrate it fails to recognize an important class of phishing URLs that contain popular brand names concatenated with one or more phishy words. Consequently, we explore various alternative feature extraction techniques to improve the robustness of classifiers. Specifically, our contributions are as follows:

1. We find that the conventional bag-of-words (BoW) feature extraction technique, based solely on special characters (‘/’, ‘?’, ‘.’, ‘=’, ‘_’, ‘&’ and ‘-’), is not robust enough to detect all types of phishing URLs. For example, the

BoW features of the URL `paypal.com.secure05.xserver.prishka1.com` extracted using techniques described in [16,17] are *name* = {`paypal, com, secure05, xserver, prishka1`} and *tld* = {`com`}. A classifier trained on these conventional BoW features correctly predicts the URL as phishing due to the high frequency of the words `paypal` and `com` in the hostname portion of the phishing URLs dataset. However, the classifier fails to predict the URL `paypalhelpservice.simdif.com` as phishing, based on the BoW features: *name* = {`paypalhelpservice, simdif`} and *tld* = {`com`}, as the tokens `paypalhelpservice` and `simdif` do not appear in the phishing dataset.

2. Therefore, to overcome these limitations of conventional bag-of-words (BoW) features, we explore other feature extraction techniques based on word segmentation and $n-$grams. We also build and use a phishy-list to recognize phishing URLs containing brand names along with phishy words.
   (a) In the word segmentation technique, we first split the entire URL string using special characters and then apply word segmentation algorithm on each token to extract segmented bag-of-words features. We refer to this feature set as SBoW. We also make distinction between words appearing in the different parts of the URL. For example, the SBoW features of the URL `paypalhelpservice.simdif.com` are *name* = {`paypal, help, service, simdif`} and *tld* = {`com`}.
   (b) In the $n-$grams technique, we split the URL string using special characters and then extract tri-grams from each resulting token. We refer to this feature set as bag-of-$n$grams (BoN). Again, we make distinction between $n-$grams appearing in the different parts of the URL. For instance, the BoN features of the URL `paypalhelpservice.simdif.com` are *name* = {`pay, ayp, ypa, pal, hel, elp, ..., dif`} and *tld* = {`com`}.
   (c) Phishing URLs often contain several words such as `login, secure, help` and `update` which are indicative of phishing attacks. Based on this observation, we retrieve popular tokens from the phishing dataset and create a phishy-list (PL) of these words. We check whether any phishy word appears in the URL and use it as a binary feature in conjunction with BoW features. We refer to this feature set as BoW-PL. For example, if the word `help` is present in the phishy-list, then the BoW-PL features of the URL `paypalhelpservice.simdif.com` are *name* = {`paypalhelpservice, simdif`}, *tld* = {`com`} and *phishy-list* = 1.
3. We evaluate the efficacy of all proposed feature sets on a dataset of 100,000 URLs obtained from PhishTank and DMOZ websites. We find that, a classifier trained on SBoW, phishy-list and numerical features (*e.g.*, URL length) outperforms classifiers trained on other feature sets.

2 Related Work

In this section, we give a brief overview of different anti-phishing countermeasures proposed in the literature. These countermeasures are broadly classified into three categories: *make things invisible*, so that users can focus on their task

instead of worrying about phishing attacks; *provide better interfaces* that assist users in detecting phishing attacks; and *train users* to proactively recognize and counter phishing attacks [13].

2.1 Making Things Invisible

Various phishing detection techniques that rely on URL based features or page based features fall under the make things invisible category. First, we describe URL based detection techniques (which is the topic of this paper) in detail followed by page based detection techniques.

URL Based Detection. In [12], Garera *et al.* identified four distinct categories of URL obfuscation techniques that the attackers use to mount phishing attacks. Further, to identify these phishing URLs, they proposed 18 different features including those based on Google infrastructure such as page rank and page quality. They determined the weight of each feature using a logistic regression model trained on a dataset of approximately 2500 URLs. McGrath *et al.* [20] performed a comparative analysis of phishing and non-phishing URLs and found that phishing URLs and domains have very different lengths and character distributions compared to non-phishing URLs and domains. As a consequence, the features based on URL length and domain length were successfully employed in classification models constructed in the subsequent studies.

Ma *et al.* [17] described a phishing detection approach that uses (a) lexical features extracted from URL names such as URL length and bag-of-words (BoW), and (b) external features acquired from queries to remote servers such as whois lookup. They examined the performance of several batch based learning algorithms on a dataset of 35,500 URLs and found that the use of lexical features achieved similar classification accuracy without incurring the overhead of querying remote queries. Later, Le *et al.* [16] performed a focused study on evaluating a classifier trained only on lexical features vs. a classifier trained on full features (lexical + external) to detect phishing attacks. They found that the performance of a classifier trained with only lexical features was similar to the one trained with full features. Their results were based on around 14,000 phishing URLs. We note that both approaches [16,17] extract BoW features by tokenizing the URL string using special characters ('/', '?', '.', '=', '_', '&' and '-') and make distinction between tokens that appear in the domain name, the top level domain, the directory, and the file extension. An extensive survey of phishing detection techniques that rely on URL based features can be found in [23]. Recently, researchers have also proposed the usage of deep neural networks for feature extraction and classification of malicious URLs [30].

In this work, we focus on improving the detection capabilities of lexical features based classifiers. We explore various lexical features based on word segmentation and $n-$grams. Further, we also construct and use a phishy-list of phishy words. In [28], Wang *et al.* explored the use of a word segmentation algorithm to improve the detection of malicious domains containing brand names concatenated with one or more phishy words. They applied the word segmentation

algorithm only on the domain portion of the URL string. However, we observed that phrases containing popular brand names and phishy words appear not only in the domain, but also in other parts of phishing URLs such as subdomain and path. Therefore, in our approach, we first tokenize the entire URL string using special characters ('/', '?', '.', '=', '_', '&' and '-') and then apply a word segmentation algorithm on each token. Further, we distinguish between tokens appearing in the hostname, tld, directory, file name and arguments portion of the URL string. Recently, Verma *et al.* [27] explored the efficacy of unigrams, bigrams and trigrams features and found that classifiers trained on $n-$gram features achieved a higher classification accuracy. While they extract $n-$grams from the URL string directly (without tokenizing), we first tokenize the URL string and then extract $n-$grams from each resulting token. In addition, we make distinction among $n-$grams belonging to hostname, tld, directory, file name and arguments, whereas they do not. We evaluate the effectiveness of each of these feature sets by training a classifier on a large dataset of 100,000 URLs.

Page Based Detection. Other phishing detection techniques rely on page based features. Zhang *et al.* [31] developed a novel content based approach called CANTINA that uses TF-IDF information retrieval algorithm to extract features from the web page. Their evaluation showed that CANTINA achieved a true positive rate of approximately 95%. Whittaker *et al.* [29] described the design of the Google's proprietary machine learning classifier that uses a variety of features such as lexical features, external features, Google Page Rank, and features extracted from the page content, to detect phishing websites. Their approach also achieved a true positive rate of around 95% and a false positive rate of 0.1%. Ardi *et al.* [7] proposed an approach that uses cryptographic hashing of each web page's Document Object Model (DOM) to detect phishing attacks. Their approach yielded a zero false positive rate.

2.2 Better Interfaces and Training

Numerous studies show that users do not pay attention to the security indicators in the browsers [6,10] nor do they adhere to the browser warning messages [11]. Although, modern web browsers have improved the design of their warning pages, many users still struggle to understand and therefore, disregard browser warning messages [22]. As a consequence, researchers have explored various training methods to teach users about the importance of various security indicators and to recognize phishing attacks. For instance, numerous educational games have been developed to educate users about phishing URLs [8,9,24] which mainly focus on teaching users about different URL obfuscation techniques as identified by Garera *et al.* [12]. Although, training users to recognize phishing attacks could complement the machine based phishing detection methods, the actual benefits of using these training techniques in the real world is not yet known.

3 Approach

In this section, we first describe the phishing and valid URL datasets used in our evaluation. Later, we discuss the pros and cons of the traditional lexical features [16,17] and describe various alternative lexical features. Finally, we give a brief overview of the logistic regression model employed to recognize phishing URLs.

3.1 Datasets

Phishing URL Dataset. PhishTank [4], a community-driven phishing URL submission and verification system operated by OpenDNS, is one of the most widely used phishing data source for training URL based classifiers [16–18,27]. A suspicious URL is marked as phish if it is voted by at least two other members of the community. The data submitted to PhishTank is available free of cost to everyone through the PhishTank's website and API. We scraped 55,000 unique verified phishing URLs from the PhishTank website during January 2019.

Valid URL Dataset. DMOZ [2] is a large open human-edited directory of the web containing over five million URLs organized hierarchically in over one million categories. It is one of the most popular source to obtain legitimate URLs [16–18,27] and contains websites from diverse categories such as arts, business, news and sports. We randomly crawled 55,000 unique URLs from the DMOZ website during January 2019.

After the data collection phase, we performed data sanitization and removed all URLs with invalid syntax. Since URL based classifiers require lexical features for predicting a label, we filtered out all short URLs from both datasets. There were no short URLs in the valid dataset, however there were about 2,000 short URLs in the phishing dataset belonging to 20 different URL shortening services. We also replaced %xx escapes in the URL with their single character equivalent, *e.g.,* %20 is replaced with *space* and %2D is replaced with *hyphen*. From the remaining URLs, we randomly chose a subset of 50,000 URLs in each of the datasets.

3.2 Features

In [12], Garera *et al.* identified four prominent URL obfuscation techniques used by the attacker. These are as follows:

- **Type I.** Obfuscation the host with an IP address: In this attack, the hostname contains an IP address and the organization being phished is placed in the path.
- **Type II.** Obfuscating the host with another domain: In this attack, the URL's hostname contains a valid looking domain name and the organization being phished is placed in the path.
- **Type III.** Obfuscating with large hostnames: In this attack, the organization being phished is present in the subdomain part of the URL.

– **Type IV.** Domain unknown or misspelled: In this attack, the domain name is misspelled or there is no apparent relationship between the organization being phished and the domain name.

Table 1. Commonly used URL obfuscation techniques illustrated using PayPal brand. The first four obfuscation techniques were identified by Garera *et al.* [12] while the Type V obfuscation was identified by Kintis *et al.* [15].

Category	Description	Examples
Type I	IP address	`http://51.77.145.33/www.paypal.com.webapps.mpp.account-selection/` `http://159.203.6.191/servicepaypal/`
Type II	Brand in path	`http://kannadamatinee.com/www.paypal.com.us/myaccount/signin` `http://a0243562.xsph.ru/servicePayPal/C/`
Type III	Brand in subdomain	`http://paypal.com.secure05.xserver.prishka1.com/` `https://paypalhelpservice.simdif.com/`
Type IV	Misspelled brand or unrelated domain	`http://paypa1.com` `http://bnkp-bdg.com/login`

Type V	Brand in domain	`http://paypal-account-limit-remove-com.ga/`
		`http://ssl-paypalupdate.com/success`
		`http://paypalnow.de/signin.htm`

Recently, Kintis *et al.* [15] identified a potent URL obfuscation technique known as *combosquatting* in which the organization being phished is present in the domain along with one or more words. We refer to this obfuscation technique as Type V. Table 1 provides illustrative examples for each of these obfuscation techniques.

To improve the classification accuracy of different obfuscating URLs, researchers [16,17] extracted two types of lexical features from the URL name: *bag-of-words* (BoW) features and *numerical* features. Originally researchers focused on detecting only the first four obfuscation techniques proposed by Garera *et al.* [12] as the Type V obfuscation is a more recent one. However, we find that some of these features are also useful in detecting Type V obfuscation. Now, we describe pros and cons of each of these features.

Bag-of-Words (BoW). The bag-of-words features are conventionally extracted by splitting the URL string into multiple tokens using special characters (‘/’, ‘?’, ‘.’, ‘=’, ‘_’, ‘&’ and ‘-’) [16,17]. Each resulting token constitutes a binary feature, the value of the feature is one if the token is present in the URL, otherwise it is zero. Further, a distinction is made among tokens appearing in the hostname, tld, directory, file name and the argument part of the URL, *i.e.*, the same word appearing in different parts of the URL is treated as a different binary feature. The main purpose of using positional bag-of-words (BoW) features is to detect Type I, Type II and Type III obfuscation techniques where the organization being phished (*e.g.*, `paypal`) or tld (*e.g.*, `com`) or phishy words (*e.g.*, `account`) appear in unexpected parts of the URL. For instance, the word `com` is more likely to appear in the tld part of the URL, however, if it appears in either subdomain or path, then the URL is a potential phish.

Table 2. Examples of BoW, SBoW, BoN and BoW-PL features

Type I URL	159.203.6.191/servicepaypal/
BoW	*name* = {159, 203, 6, 191}, *tld* = {}, *dir* = {servicepaypal}
SBoW	*name* = {159, 203, 6, 191}, *tld* = {}, *dir* = {service, paypal}
BoN	*name* = {159, 203, 6, 191}, *tld* = {}, *dir* = {ser, erv, rvi, vic, ice, . . ., pal}
BoW-PL	*name* = {159, 203, 6, 191}, *tld* = {}, *dir* = {servicepaypal}, *phishy-list* = 1
Type II URL	a0243562.xsph.ru/servicePayPal/C/
BoW	*name* = {a0243562, xsph}, *tld* = {ru}, *dir* = {servicepaypal, c}
SBoW	*name* = {a0243562, xsph}, *tld* = {ru}, *dir* = {service, paypal, c}
BoN	*name* = {a02, 024, 243, 435, . . ., sph}, *tld* = {ru}, *dir* = {ser, erv, rvi, . . ., c}
BoW-PL	*name* = {a0243562, xsph}, *tld* = {ru}, *dir* = {servicepaypal, c}, *phishy-list* = 1
Type III URL	paypalhelpservice.simdif.com
BoW	*name* = {paypalhelpservice, simdif}, *tld* = {com}
SBoW	*name* = {paypal, help, service, simdif}, *tld*= {com}
BoN	*name* = {pay, ayp, ypa, pal, . . ., dif}, *tld* = {com}
BoW-PL	*name* = {paypalhelpservice, simdif}, *tld* = {com}, *phishy-list* = 1
Type V URL	ssl-paypalupdate.com/success
BoW	*name* = {ssl, paypalupdate}, *tld* = {com}, *dir* = {success}
SBoW	*name* = {ssl, paypal, update}, *tld* = {com}, *dir* = {success}
BoN	*name* = {ssl, pay, ayp, . . ., ate}, *tld* = {com}, *dir* = {suc, ucc, cce, ces, ess}
BoW-PL	*name* = {ssl, paypalupdate}, *tld* = {com}, *dir* = {success}, *phishy-list* = 1

We observed that although a classifier trained on conventional BoW features performs well in many cases, it fails to recognize phishing URLs that combine a popular brand with one or more words. Table 2 shows BoW features for URLs belonging to different obfuscation techniques. Since the tokenization procedure employed in extracting BoW features rely only on special characters, long phrases such as `servicepaypal`, `paypalhelpservice` and `paypalupdate` remain unsegmented in BoW features. The prediction scores of the Type I and Type II URLs could be improved if the token `servicepaypal` in the directory is further segmented into individual words `service` and `paypal`. Similarly, the Type III URL is more likely to be classified correctly, if the token `paypalhelpservice` in the subdomain is further segmented into words `paypal`, `help` and `service`. Therefore, we explore different lexical features based on the word segmentation and $n-$grams, and use a list of phishy words to improve the prediction of phishing URLs.

Segmented Bag-of-Words (SBoW). We use word segmentation based technique to extract BoW features from the URL string which are more robust against combosquatting URLs. In this technique, we first extract tokens from the URL string using special characters ('/', '?', '.', '=', '_', '&' and '-'). Subsequently, we apply a word segmentation algorithm on each extracted token to recover the individual words. We use Python's WordSegment module [5] for word segmentation which is based on code by Peter Norvig that uses *Google Web Trillion Word Corpus* [21]. Table 2 shows SBoW features for different obfuscated URLs. For example, after applying the word segmentation algorithm, the token `paypalhelpservice` in the Type III URL is now further divided into a set of three words {`paypal, help, service`}.

Bag-of-ngrams (BoN). We also explore $n-$gram based features to improve the detection of phishing URLs containing brand names and words. In this technique, we first extract tokens from the URL string using special characters ('/', '?', '.', '=', '_', '&' and '-'). Subsequently, we extract tri-grams from each token and use them as binary features. BoN features for four different obfuscated URLs are given in Table 2. For example, the trigrams of the token `paypalhelpservice` are {`pay, ayp, ypa, pal, alh, lhe, hel, elp, lps, pse, ser, erv, rvi, vic, ice`}.

Phishy-List (PL). In this technique, we construct a list of popular phishy tokens by analysing URL domains in the phishing dataset. We discard all tokens with *length* ≤ 3 as they contain common URL parts such as `com` and `org`. We remove organization name tokens like `paypal` to keep our phishy-list brand agnostic. The resulting list contains 105 popular words (frequency ≥ 20) indicative of phishing attacks. We refer to this list as *new-PL* (provided in Appendix A). Few examples of popular phishy words are `secure, login, account, update, verify` and `service`. We use this phishy-list as a binary feature and check whether any of the phishy tokens appear in the URL. The main purpose of using the phishy-list is to detect phishing URLs that contain brand names concatenated with popular phishy words. The phishy-list feature was also used in [16] (referred as blacklist feature) to address Type IV obfuscation. However, their phishy-list was small and contained only 12 words: `confirm, account, banking, secure, ebayisapi, webscr, login, signin, paypal, free, lucky` and `bonus`. We refer to this list as *legacy-PL*. We emphasize that our phishy-list is large and contains 105 popular brand agnostic phishy tokens.

Numerical Features. We also extract various numerical features as described by Le *et al.* [16]. First, the URL string is broken into four parts: domain, directory, file name and arguments. Subsequently, numerical features in each of these parts are retrieved. Table 3 shows different numerical features of a URL.

Table 3. Numerical features of a URL

URL	`paypal-billing.my-profilemanage.com/login/myaccount/webscr_login/index.php ?cmd_=login-submit`
Features	len = 92, n_dot = 3
Hostname	`paypal-billing.my-profilemanage.com`
Features	len = 35, IP = 0, port = 0, n_token=5, n_hyphen = 2, max_len = 13
Directory	`/login/myaccount/webscr_login/`
Features	len = 30, n_subdir = 3, max_len = 9, max_dot = 0, max_delim = 1
Filename	`index.php`
Features	len = 9, n_dot = 1, n_delim = 0
Arguments	`?cmd_=login-submit`
Features	len = 18, n_var = 1, max_len = 6, max_delim = 2

1. *URL related features.* These features include the length of the URL and the number of dots in the URL. These features are used to address Type II obfuscation.

2. *Domain related features.* These features include the length of the domain name, the number of tokens in the domain name, the number of hyphens in the domain name, the length of the longest token and whether an IP address or a port number is present in the domain name. Although, these features are used to address Type I and Type III obfuscation techniques, these features particularly the number of hyphens can also detect few instances of Type V obfuscation.
3. *Directory related features.* These features include the length of the directory, the number of sub-directory tokens, the length of the longest sub-directory token, and the maximum number of dots and other delimiters ('_' and '-') used in a sub-directory token. These features are proposed to address the Type II obfuscation technique.
4. *File name related features.* These features include the length of the file name, and the number of dots and other delimiters ('_' and '-') used in the file name. These features also used to address Type II obfuscation.
5. *Argument related features.* These features include the length of the arguments, the number of variables, the length of the longest variable value, and the maximum number of delimiters ('.', '_' and '-') used in a value.

Thus, a total of 20 numerical features are extracted from different parts of a URL. Table 4 compares the numerical features in valid and phishing datasets. URLs in the phishing dataset are much longer (more than 2x times) and contain more special characters (hyphen, dot) as compared to URLs in the valid dataset. We use these numerical features along with BoW, SBoW and BoN features.

Table 4. Analysis of numerical features in phishing and valid datasets

URL	len	n_dot	blacklist			
Valid	31.31	2.16	0.01			
Phishing	73.45	2.44	0.32			
Hostname	len	IP	port	n_token	n_hyphen	max_len
Valid	18.77	0	0	3.05	0.09	10.07
Phishing	20.99	0.02	0	2.71	0.35	11.17
Directory	len	n_subdir	max_len	max_dot	max_delim	
Valid	2.36	0.31	1.59	0	0.04	
Phishing	20.92	2.09	10.44	0.13	0.34	
File	len	n_dot	n_delim			
Valid	1.76	0.1	0.06			
Phishing	7.02	0.47	0.11			
Arguments	len	n_var	max_len	max_delim		
Valid	0.2	0.02	0.08	0		
Phishing	15.05	0.41	5.78	0.16		

3.3 Logistic Regression for URL Classification

The problem of phishing detection is formulated as a binary classification task with two classes: *phishing* (positive class) and *valid* (negative class). We use logistic regression as it is computationally efficient and improves the performance by retaining only the relevant features. It is a simple parametric model where URLs are classified based on their distance from hyperplane decision boundary. In the binary classification task, we are given $\mathcal{M}$ training instances $\{x_1, x_2, \ldots, x_M\}$, where each x_i is a N dimensional feature vector and $y_i \in \{0, 1\}$ is a class label associated with sample x_i. Logistic regression models the probability distribution of the class label y, given a feature x as follows:

$$p(y = 1|x; \theta) = \sigma(\theta^T x + b) = \frac{1}{1 + \exp^{-(\theta^T x + b)}} \tag{1}$$

where, $\theta \in \mathbb{R}^N$ and bias b are the parameters of the logistic regression model, and $\sigma(\cdot)$ is the sigmoid function defined as $\sigma(z) = 1/(1 + \exp^{-z})$. This sigmoid function $\sigma(\cdot)$ interprets the distances as probabilities of positive and negative labels.

We train the logistic regression model using maximum likelihood estimation with l_1 regularization. We estimate the weight vector θ and bias b by maximizing the objective function:

$$\mathcal{L}(\theta, b) = \sum_{i=1}^{M} \log p(y_i|x_i) - \lambda \sum_{j=1}^{N} |\theta_j| \tag{2}$$

The first term in Eq. 2 computes the conditional log-likelihood that the model predicts correct label for all the samples in the training set. The second term in the equation penalizes large magnitude values in the weight vector θ. This is known as l_1 norm regularization and has many beneficial properties over SVM and Naive Bayes estimators while working with large feature dimensions. (i) It serves as a measure against overfitting; (ii) it encourages sparse solutions in which many elements of the weight vector θ are *exactly* zero (iii) it also helps in feature selection by retaining only the most relevant features. Due to these benefits, the logistic regression classifier has been widely used to develop various anti-phishing solutions in the past [12,17,28].

4 Results and Discussion

Now, we evaluate and compare the efficacy of classifiers trained on various feature sets described in Sect. 3. Specifically, we investigate how different feature extraction techniques help in distinguishing phishing URLs from valid URLs. To this end, we train logistic regression classifiers on different feature sets and report their misclassification rate (MCR) and false negative rate (FNR). MCR measures the rate of incorrectly detected valid and phishing instances in relation

to all instances, whereas false negative rate (FNR) measures the rate of phishing instances that are incorrectly detected as valid in relation to all phishing instances. Specifically,

$$MCR = \frac{N_{P\to V} + N_{V\to P}}{N_{P\to P} + N_{P\to V} + N_{V\to V} + N_{V\to P}} \tag{3}$$

and,

$$FNR = \frac{N_{P\to V}}{N_{P\to P} + N_{P\to V}} \tag{4}$$

where $N_{P\to P}$ is the number of phishing URLs correctly identified as phishing, $N_{V\to V}$ is the number of valid URLs correctly identified as valid, $N_{P\to V}$ is the number of phishing URLs incorrectly identified as valid and $N_{V\to P}$ is the number of valid URLs incorrectly identified as phishing. Our objective is to minimize both MCR and FNR. For training, we randomly select a subset of 80,000 URLs and use the remaining 20,000 URLs for testing. In our classification tasks, we consider phishing URL as positive class and valid URL as negative class.

We divide our experiments into three parts. Firstly, we investigate the effectiveness of three logistic regression classifiers trained on different bag-of-X representations, namely BoW [16,17], SBoW and BoN. Secondly, we compare the effectiveness of two phishy-lists, legacy-PL [16] and our proposed new-PL. Finally, we determine the potency of combining these different features with numerical features. The list of feature sets used in our classification experiments along with their corresponding MCR and FNR are given in Table 5. The table also shows the total number of extracted features in each feature set, the number of retained (non-zero) features, the number of retained features with positive (+ve) and negative (−ve) weights, and FNR reduction (FNR-Red) with respect to the baseline classifier (trained only on BoW features).

Table 5. Performance of classifiers trained with different feature sets based on MCR, FNR and reduction in FNR. We also report the number of features in each feature set, the number of relevant features, and features with +ve and −ve coefficients. Note that *num* represents numerical features.

Feature set	#Features	#Relevant	+ve	−ve	MCR(%)	FNR(%)	FNR-Red(%)
BoW (baseline)	107,277	2,240	1,767	473	5.04	7.87	–
BoN	108,038	3,987	2,621	1,366	4.18	5.29	32.78
SBoW	88,930	2,692	1,941	751	4.07	5.57	29.22
BoW+legacy-PL	107,278	2,201	1,728	473	5.02	7.84	0.38
BoW+new-PL	107,278	1,885	1,426	459	4.23	5.70	27.57
SBoW+new-PL	88,931	2,318	1,619	699	3.63	4.59	41.67
BoW+legacy-PL+num [16]	107,298	1,809	1,199	610	4.05	5.72	27.31
BoW+new-PL+num	107,298	1,604	1,048	556	3.70	4.83	38.62
BoN+num	108,058	3,428	1,569	1,859	3.44	4.25	45.99
SBoW+new-PL+num	**88,951**	**2,124**	**1,354**	**770**	**3.22**	**4.10**	**47.90**

Bag-of-X. Our experimental results show that a logistic regression classifier trained only on conventional BoW features [16,17] (tokens extracted using spe-

cial characters) yielded a MCR of 5.04%. However, classifiers trained on BoN (tri-gram) features and SBoW features (tokens extracted using special characters and word segmentation) reduced the MCR to 4.18% and 4.07% respectively. A deeper analysis of the results show that SBoW and BoN features are more robust (low FNR) against all types of phishing URLs as compared to BoW features. Few examples of such URLs are illustrated in Table 6, where the classifier trained on BoW features misclassified the phishing URL as valid whereas classifiers trained on SBoW and BoN did not. For instance, based on BoW features, the Type II URL `al-cap.com/vvb/chaseonline2018` (a spoof of US based *Chase* bank) is labelled as phish with a probability of 0.42. Note that in this case, only the tld token `com` is determined as a relevant feature (and that too negative) by the logistic regression classifier, whereas other tokens such as `cap` and `chaseonline2018` are simply ignored since their corresponding weights are zero. The SBoW features on the other hand extract relevant tokens `chase` and `online` from the phrase `chaseonline2018` which are determined as positive features by the classifier. As a consequence, the URL is classified as phishing with a very high probability (0.99). The classifier trained on BoN features performed similarly to that trained on SBoW features. Consequently, when compared to the BoW model, FNR of BoN reduced by 32.78% and FNR of SBoW reduced by 29.22%. Although FNR of BoN is slightly less than FNR of SBoW, the number of features retained in the BoN model (3,987) is almost 1.5 times more than those retained in the SBoW model (2,692). Therefore, the model trained using SBoW features is simpler than the model trained using BoN features and exhibit comparable performance.

Table 6. Illustrative examples demonstrating the effectiveness of logistic regression classifiers trained on SBoW and BoN features over classifier trained on BoW features.

URL	Features	+ve features	−ve features	Prob
Type II	BoW	–	com	0.42
al-cap.com/vvb/chaseonline2018	SBoW	online, chase	com	**0.99**
	BoN	cha, has, ase, lin, nli	com, eon, ine	**0.99**
Type III	BoW	–	blogspot, com	0.19
facebookloginconfirmation.	SBoW	facebook, login, confirmation	blogspot, com	**0.99**
blogspot.com	BoN	fac, ceb, ebo, boo, con, ...	log, com, ...	**0.98**
Type IV	BoW	account	www, com	0.28
www.amzaon-account.com/	SBoW	am, on, account	www, com	**0.68**
	BoN	cco, amz, acc, zao	oun, com, www	**0.93**
Type V	BoW	–	com	0.42
google1mail.com/mi-cuenta	SBoW	mail, cuenta, google	com	**0.95**
	BoN	nta, mai, ail, cue, ogl	oog, gle, ent, com	**0.90**

Phishy-Lists. We trained two logistic regression classifiers to determine the quality of two phishy lists, legacy-PL and new-PL. Both classifiers were trained on conventional BoW features, the only difference was that the first classifier considered legacy-PL whereas the second classifier considered new-PL. We note that

phishy-list is a binary feature, where we check if any of the words in the phishy-list appear in the URL. Therefore, the total number of features used for training two classifiers were same (107,278). However, after training, we found that the first classifier with legacy-PL retained 2,201 features, whereas the other classifier that used new-PL retained only 1,885 features. Also, MCR of the first classifier with legacy-PL was 5.02%, slightly better than the BoW features (5.04%). Further, there was only a miniscule reduction of 0.38% in FNR. On the other hand, MCR of the second classifier with new-PL was 4.23% and its FNR reduced by 27.57%. Therefore, the use of new-PL resulted in simple model and improved accuracy. Replacing the BoW features with SBoW features and using new-PL, reduced the MCR to 3.63%. Also, its FNR decreased by 41.67%. Hence, as new-PL outperformed legacy-PL, we conduct the remaining experiments using new-PL only.

Full Feature Set. From the experiments above, it can be seen that BoX features, alone, perform very well in classifying phishing URLs. However, these BoX features are not always enough to model the unseen URLs. Hence, we require a set of orthogonal features that complement the BoX features. Therefore, we also consider 20 numerical features to make classifiers more robust. The performance of classifiers trained on the following combination of feature sets (BoX, phishy-list, numerical) is shown in Table 5.

1. BoW + numerical + legacy-PL (state-of-the-art): In this we implemented a logistic regression classifier based on features proposed in [16]. These state-of-the-art features resulted in a MCR of 4.05%. Further, we observed a FNR reduction of 27.31% against the baseline BoW features. However, as shown in Table 5, this feature set is outperformed by all other full feature sets in terms of MCR as well as FNR. Further, the classifier trained only on SBoW and new-BL features (with MCR 3.63% and FNR reduction of 41.67%) performed better than the current classifier that used BoW features, legacy-PL as well as numerical features.
2. BoW + numerical + new-PL: Here, instead of using the legacy-PL consisting of 12 phishy words [16], we used a larger new-PL consisting of 105 brand agnostic phishy words. We obtained a MCR of 3.70% and FNR reduction of 38.62%, both better than the state-of-the-art features proposed in [16]. However, these MCR and FNR are still higher than those achieved using only SBoW and new-PL features.
3. BoN + numerical: After training a logistic regression classifier on tri-gram features and numerical features, we obtained a MCR of 3.44% and FNR reduction of 45.99% on the test set. This is the second best feature set among all other feature sets.
4. SBoW + numerical + new-PL: Here, we used SBoW features, numerical features as well as new-BL. We obtained a MCR of 3.22% and FNR reduction of 47.90%, which is the lowest among all feature sets.

5 Conclusion and Future Work

In this paper, we demonstrated that a classifier trained on conventional lexical features fails to recognize phishing URLs that contain a brand name concatenated with one or more phishy words (*e.g.*, `paypalhelpservice.simdif.com`). To overcome these limitations, we explored different bag-of-X representations including bag-of-words (BoW), segmented bag-of-words (SBoW) and bag-of-ngrams (BoN). We found that a logistic regression classifier trained on SBoW features resulted in lower misclassification rate (MCR) as well as lower false negative rate (FNR) when compared with BoW features. Further, SBoW features yielded a simpler model when compared with BoN features. We also proposed a new phishy-list consisting of 105 brand agnostic words suggestive of phishing attacks and compared its performance against the legacy phishy-list [16]. The results of our experiments suggest that the new phishy-list not only improved the detection of phishing URLs, but also resulted in a simpler model. Further, we found that combining numerical features with SBoW features and new phishy-list outperformed all other combinations of feature sets used for phishing detection.

The feature extraction techniques proposed in this paper are well suited for detecting Type III (brand in subdomain), Type V (brand in domain), and Type II (brand in path) phishing URLs. But still, there is a lot to be desired for Type IV phishing URLs which are composed of unrelated or misspelled domains. We plan to explore the techniques to counter these URLs in our future work.

Appendix A

The phishy-list consisting of 105 words extracted from the phishing dataset is given below:

{`limited, securewebsession, confirmation, page, signin, team, sign, access, protection,active, manage, redirectme, http, secure, customer, account, client, information, recovery, verify, secured, busines, refund, help, safe, bank, event, promo, webservis, giveaway, card, webspace, user, notify, servico, store, device, payment, webnode, drive, shop, gold, violation, random, upgrade, webapp, dispute, setting, banking, activity, startup, review, email, approval, admin, browser, webapp, billing, advert, protect, case, temporary, alert, portal, login, servehttp, center, client, restore, secure, blob, smart, fortune, gift, server, security, page, confirm, notification, core, host, central, service, account, servise, support, apps, form, info, compute, verification, check, storage, setting, digital, update, token, required, resolution, ebayisapi, webscr, login, free, lucky, bonus`}

References

1. APWG, February 2019. http://docs.apwg.org/reports/apwg_trends_report_q3_2018.pdf
2. DMOZ, February 2019. http://dmoz-odp.org/
3. Google Safe Browsing, February 2019. https://safebrowsing.google.com/
4. PhishTank, February 2019. https://www.antiphishing.org/resources/apwg-reports/
5. Python Word Segmentation, February 2019. http://www.grantjenks.com/docs/wordsegment/
6. Alsharnouby, M., Alaca, F., Chiasson, S.: Why phishing still works: user strategies for combating phishing attacks. Int. J. Hum.-Comput. Stud. **82**, 69–82 (2015)
7. Ardi, C., Heidemann, J.: Auntietuna: personalized content-based phishing detection. In: Proceedings of the NDSS Workshop on Usable Security. The Internet Society, San Diego, California, USA, February 2016. http://www.isi.edu/%7ejohnh/PAPERS/Ardi16a.html
8. Canova, G., Volkamer, M., Bergmann, C., Reinheimer, B.: NoPhish app evaluation: lab and retention study. Internet Society, USEC (2015)
9. CJ, G., Pandit, S., Vaddepalli, S., Tupsamudre, H., Banahatti, V., Lodha, S.: Phishy - a serious game to train enterprise users on phishing awareness. In: Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts, CHI PLAY 2018, pp. 169–181. ACM, New York (2018). https://doi.org/10.1145/3270316.3273042
10. Dhamija, R., Tygar, J.D., Hearst, M.: Why phishing works. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2006, pp. 581–590. ACM, New York (2006). https://doi.org/10.1145/1124772.1124861
11. Felt, A.P., et al.: Improving SSL warnings: comprehension and adherence. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 2893–2902. ACM, New York (2015). https://doi.org/10.1145/2702123.2702442
12. 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, WORM 2007, pp. 1–8. ACM, New York (2007). https://doi.org/10.1145/1314389.1314391
13. Hong, J.: The state of phishing attacks. Commun. ACM **55**(1), 74–81 (2012). https://doi.org/10.1145/2063176.2063197
14. Khonji, M., Iraqi, Y., Jones, A.: Phishing detection: a literature survey. IEEE Commun. Surv. Tutor. **15**(4), 2091–2121 (2013). https://doi.org/10.1109/SURV.2013.032213.00009
15. Kintis, P., et al.: Hiding in plain sight: a longitudinal study of combosquatting abuse. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, pp. 569–586. ACM, New York (2017). https://doi.org/10.1145/3133956.3134002
16. Le, A., Markopoulou, A., Faloutsos, M.: PhishDef: URL names say it all. In: 2011 Proceedings IEEE INFOCOM, pp. 191–195, April 2011. https://doi.org/10.1109/INFCOM.2011.5934995
17. Ma, J., Saul, L.K., Savage, S., Voelker, G.M.: Beyond blacklists: learning to detect malicious web sites from suspicious URLs. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, pp. 1245–1254. ACM, New York (2009). https://doi.org/10.1145/1557019.1557153

18. Marchal, S., François, J., State, R., Engel, T.: Phishstorm: detecting phishing with streaming analytics. IEEE Trans. Netw. Serv. Manag. **11**(4), 458–471 (2014). https://doi.org/10.1109/TNSM.2014.2377295
19. Marchal, S., Saari, K., Singh, N., Asokan, N.: Know your phish: novel techniques for detecting phishing sites and their targets. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), pp. 323–333, June 2016. https://doi.org/10.1109/ICDCS.2016.10
20. McGrath, D.K., Gupta, M.: Behind phishing: an examination of phisher modi operandi. In: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats, LEET 2008, pp. 4:1–4:8. USENIX Association, Berkeley, CA, USA (2008). http://dl.acm.org/citation.cfm?id=1387709.1387713
21. Norvig, P.: Natural Language Corpus Data: Beautiful Data, February 2019. http://norvig.com/ngrams/
22. Reeder, R.W., Felt, A.P., Consolvo, S., Malkin, N., Thompson, C., Egelman, S.: An experience sampling study of user reactions to browser warnings in the field. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI 2018, pp. 512:1–512:13. ACM, New York (2018). https://doi.org/10.1145/3173574.3174086
23. Sahoo, D., Liu, C., Hoi, S.C.: Malicious URL detection using machine learning: a survey. arXiv preprint arXiv:1701.07179 (2017)
24. Sheng, S., et al.: Anti-phishing phil: the design and evaluation of a game that teaches people not to fall for phish. In: Proceedings of the 3rd Symposium on Usable Privacy and Security, SOUPS 2007, pp. 88–99. ACM, New York (2007). https://doi.org/10.1145/1280680.1280692
25. Sheng, S., Wardman, B., Warner, G., Cranor, L., Hong, J., Zhang, C.: An empirical analysis of phishing blacklists. In: Sixth Conference on Email and Anti-Spam (CEAS), California, USA (2009)
26. Verizon: 2018 data breach investigations report, February 2019. http://www.verizonenterprise.com/resources/reports/rp_DBIR_2018_Report_en_xg.pdf
27. Verma, R., Das, A.: What's in a URL: fast feature extraction and malicious URL detection. In: Proceedings of the 3rd ACM on International Workshop on Security and Privacy Analytics, IWSPA 2017, pp. 55–63. ACM, New York (2017). https://doi.org/10.1145/3041008.3041016
28. Wang, W., Shirley, K.: Breaking bad: detecting malicious domains using word segmentation. arXiv preprint arXiv:1506.04111 (2015)
29. Whittaker, C., Ryner, B., Nazif, M.: Large-scale automatic classification of phishing pages. In: NDSS 2010 (2010). http://www.isoc.org/isoc/conferences/ndss/10/pdf/08.pdf
30. Yang, W., Zuo, W., Cui, B.: Detecting malicious urls via a keyword-based convolutional gated-recurrent-unit neural network. IEEE Access **7**, 29891–29900 (2019). https://doi.org/10.1109/ACCESS.2019.2895751
31. Zhang, Y., Hong, J.I., Cranor, L.F.: Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, pp. 639–648. ACM, New York (2007). https://doi.org/10.1145/1242572.1242659