NLP Based Phishing Attack Detection from URLs

Chapter *in* Advances in Intelligent Systems and Computing · March 2018

DOI: 10.1007/978-3-319-76348-4_59

CITATIONS
44

READS
5,729

3 authors, including:

Ebubekir Buber

10 PUBLICATIONS **968** CITATIONS

SEE PROFILE

Banu Diri

Yıldız Technical University

200 PUBLICATIONS **3,582** CITATIONS

SEE PROFILE

NLP Based Phishing Attack Detection from URLs

Ebubekir Buber[1], Banu Diri[1], and Ozgur Koray Sahingoz[2(✉)]

[1] Computer Engineering Department, Yildiz Techical University, Istanbul, Turkey
ebubekirbbr@gmail.com, banu@ce.yildiz.edu.tr
[2] Computer Engineering Department, Istanbul Kultur University, 34158 Istanbul, Turkey
o.sahingoz@iku.edu.tr

Abstract. In recent years, phishing has become an increasing threat in the cyber-space, especially with the increasingly use of messaging and social networks. In traditional phishing attack, users are motivated to visit a bogus website which is carefully designed to look like exactly to a famous banking, e-commerce, social networks, etc., site for getting some personal information such as credit card numbers, usernames, passwords, and even money. Lots of the phishers usually make their attacks with the help of emails by forwarding to the target website. Inexperienced users (even the experienced ones) can visit these fake websites and share their sensitive information. In a phishing attack analysis of 45 countries in the last quarter of 2016, China, Turkey and Taiwan are mostly plagued by malware with the rate of 47.09%, 42.88% and 38.98%. Detection of a phishing attack is a challenging problem, because, this type of attacks is considered as semantics-based attacks, which mainly exploit the computer user's vulnerabilities. In this paper, a phishing detection system which can detect this type of attacks by using some machine learning algorithms and detecting some visual similarities with the help of some natural language processing techniques. Many tests have been applied on the proposed system and experimental results showed that Random Forest algorithm has a very good performance with a success rate of 97.2%.

Keywords: Machine learning · Phishing attack · Random Forest Algorithm
Cyber attack detection · Cyber security

1 Introduction

In the last few decades, Internet has become the most preferred medium between the users for transferring necessary information between them. This constructs a cyberspace in the digital world, which facilitates and speeds up human life such as electronic banking, social networks, e-mail usage and electronic commerce, etc. In addition to these advantages of the Internet, it has an open anonymous infrastructure for cyber-attacks which presents serious security vulnerabilities for the users especially inexperienced ones. Cyber-attacks are causing massive loss of personal and sensitive information and even money to companies and individuals every day. The material losses are at the level of billions of dollars annually. Phishing Attacks are one of the attack types that cause

A. Abraham et al. (Eds.): ISDA 2017, AISC 736, pp. 608–618, 2018.
https://doi.org/10.1007/978-3-319-76348-4_59

the most material damage. In this attack method, the attacker (also named as phisher) opens a site which is similar to a known legal site in the Internet through his domain and tries to capture some personal and sensitive information such as username, password, bank information, personal information of the victim. This type of attack especially started with fake emails and in a study conducted by the Anti-Phishing Working Group (APWG) on 45 countries around the world. The countries which are most plagued by malware is firstly China with the rate of 47.09% (of machines are infected) and then followed by Turkey (42.88%) and Taiwan (38.98%) [1].

When users connect to this site, which is believed to be legitimate, they enter the desired information without knowing that the site is fraudulent. The attacker usually directs the user to his or her site via email. An example scenario is shown in Fig. 1. The web page visited when the link in the e-mail is actually a hidden link is shown in it. When this page is entered, it appears that the original web page is designed almost the same. Although the difference in URL address is noteworthy, for an inexperienced user, this page is a good trap.

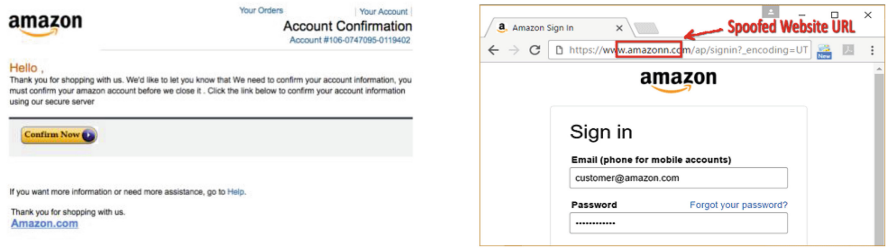


Fig. 1. Example of a mail and a web page used in phishing attack

There are many studies in the literature to detect Phishing Attacks [3]. These efforts are mainly focused on Natural Language Processing [4], Image Processing [5, 6], Machine Learning [7], Rule [8], White List - Black List [9], etc. Detection of an attack from a URL address can be a trivial task even for many experts, because an attacker can even deceive knowledgeable users by using different techniques. Simplest one is the use of rules as mentioned in [10] with similar rule format. However, rule based techniques are not satisfactory and there are many additional techniques which can be classified as depicted in Fig. 2 [2]. For this reason, it is important to detect these attacks with software support and a single approach cannot be sufficient for a good mechanism, some hybrid approaches can be preferred.

In this paper, domain names used in punctuation attacks were tried to be detected by using Natural Language Processing (NLP) techniques. We also identified some supporting features for the detection of such attacks and evaluated the effect on the performance of the proper use of these features. Specified features alone are not enough to decide whether a domain name is a cheat, but it will provide very useful information in order to make a right decision. The results obtained for this purpose should be evaluated as a result of a decision support system and should be evaluated as a parameter in the final decision.

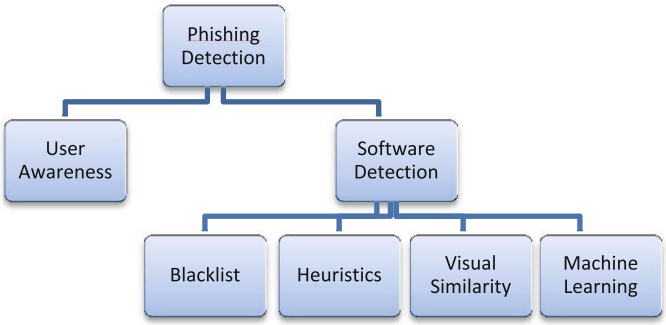


Fig. 2. An overview of phishing detection approaches

The rest of the report is organized as follows. Factors that make it difficult to identify the domain names used in Phishing attacks in the next section. Explanations about the data set used and pre-operations applied to the data set to extract the features to be used for the detection of Phishing attacks are explained in Sect. 3 and evaluations of the performed test results in Sect. 4. Finally, Conclusions are depicted.

2 URL and Attackers' Techniques

Attackers use different number of techniques to ensure that phishing attacks cannot be detected by either safety mechanisms or user. In this section these techniques and topics will be mentioned. The components in the URL (Uniform Resource Locator) structure need to be known to understand the attackers' techniques. Figure 3 shows the components found in a standard URL.

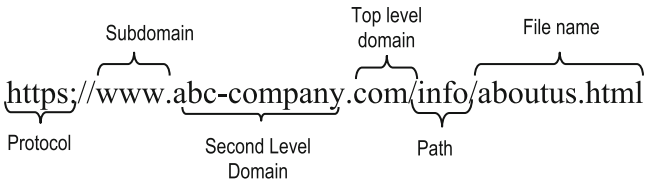


Fig. 3. URL components

The URL begins with a protocol used to access the page. The second level domain name (SLD) identifies the server hosting the web page and top level domain name specifies the domain name extension.

An attacker can purchase and use any SLD name that is not currently in use for attack purposes. This part of the URL can only be set once. An attacker can generate an unlimited number of URLs that can be used in phishing attacks by making the desired change in the Free URL field.

Because the unique part of each URL is the domain name part, cyber security companies make a lot of effort to identify this domain. When a domain name used in a

phishing attack is detected, it is easy to block access to that domain and access to an unlimited number of URLs under that domain name can be blocked.

An attacker who wishes to perform a phishing attack uses the following basic methods to increase attack performance and steal more user information: typosquatting, cybersquatting, combined word usage, and use of random characters.

3 Data Sets and Preprocessing

In order to develop a system for detecting phishing attacks, two classes of URLs are needed. These classes are Malicious URL and Legal URL. In order to detect the URLs used in phishing attacks, it is necessary to extract distinguished the features. Malicious URLs may contain known brand names or some key words. In order to be able to detect a usage in this way, brand names and keywords lists are needed. In this study, these lists were created using different sources.

The malicious URLs analyzed in this study are provided from PhishTank [11]. Yandex Search API [12] has been used to collect legal URLs. Because the malicious URLs are not allowed to have high ranking in search engines, the URLs that are ranked high obtained by sending the query words to Yandex Search API are regarded as legal URLs.

The components within a URL are separated from each other using some special characters. For example, the Domain Name and the TLD are separated from each other by a dot mark. Likewise, the domain and the subdomain are also separated by a dot. Separating within the file path is done according to the "/" sign.

Each element within the URL may contain a separation mark of its own within it. For example, "-" in the Domain in the URL "abc-company.com" made a separation. Similar usage can be done with "=, ?, &" characters in File Path Area. Before starting the URL preprocessing, each word is added to the list of words to be analyzed by extracting words that are separated from each other by a special character. The flow diagram for the data preprocessing is given in Fig. 4a.

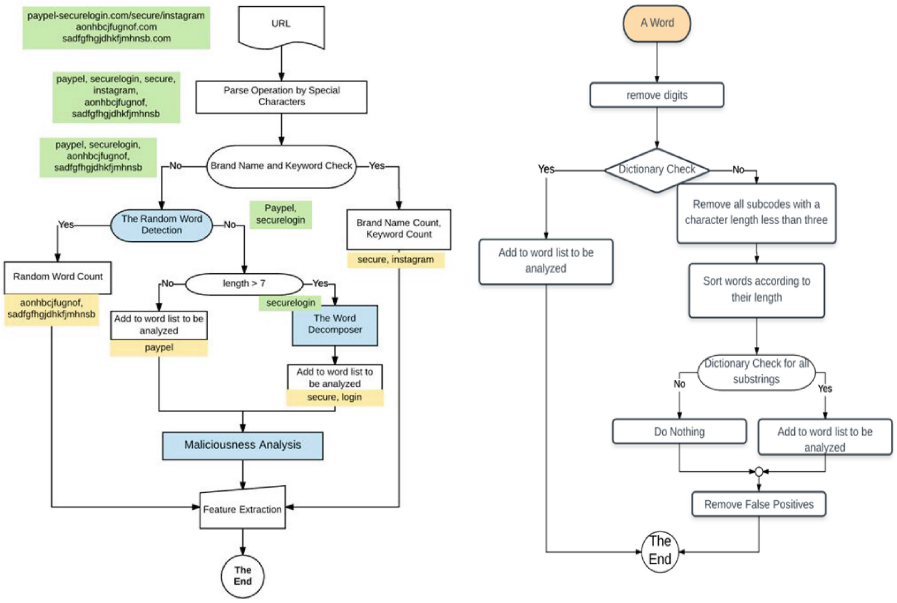
The objectives for data pre-processing can be summarized as follows:

1. To detect words with a known brand name or something similar in the URL
2. To detect keywords in the URL or something similar
3. To detect words created with random characters.

In the flowchart shown in Fig. 4b, the brand name or key words are first extracted in the words to be analyzed and the remaining words to be analyzed are given to the Random Word Detection Module.

The Random Word Detection Module (RWDM) checks whether a word is composed of random characters or not. The detected words are saved in a random word list and removed from the list of words to be analyzed.

When the words used in the phishing attacks were examined, it was determined that the length of the words with the contiguous writing was excessive. Therefore, the heuristic based threshold is determined to detect adjacent words. Words with more than 7 characters in length have been tried in Word Decomposer Module (WDM) to separate



a) Data Preprocessing Module b) Word Decomposer Module

Fig. 4. The flow diagram for the data preprocessing and word decomposer [16]

the words they contain. If the word to be analyzed has two or more words, WDM separate words. If the word has not two or more words within it, WDM return raw the word.

The words with less than 7 characters in length and the words obtained from the Word Decomposition Module were analyzed and given to the Maliciousness Analysis Module (MAM). Finally, some features related to the words analyzed have been extracted.

3.1 The Word Decomposer Module

Word Decomposer Module (WDM) is a module that returns words as separate objects if there is more than one written word in a given character string. If no words are found in the analyzed word, the main word is returned. The flow diagram for performing the decomposition process is given in Fig. 3. First, in order to start the process of separating a character string, it was checked whether the related string was found in the dictionary. A word found in the dictionary has been added to the Word List without being parsed. The designed module can distinguish the English words which are written contiguously. The Enchant [13] package, developed in Python language, is used to check if a word is found in the English dictionary. All possible substrings are extracted for the word being parsed. Main string is divided into consecutive characters. An example of the process of splitting main word into substrings is given in Fig. 5.

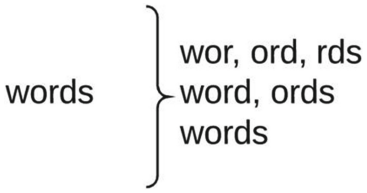


Fig. 5. Substring extraction

Extracted substrings are sorted by character length. The string with the highest character length is checked first. It is checked whether each substring is in the dictionary or not. The substrings in the dictionary are added to the word list.

There may be some false positive words in the word list obtained. For example, the "secure" word is in the list while the "cure" word is in the list. Both words are in the English dictionary and both words are in substring list. In this case, some words need to be eliminated. If the longer words contain smaller words in substring list, smaller words are eliminated on the word list to eliminate the false positive words.

It is not the case that any brand name or key word with a character length greater than 7 is given to the word parser module because the keywords are cleaned from the words to be analyzed with the control in the first stage. However, dictionary words that have a character length greater than 7 are not on the keyword list enter the WDM. These words are returned as a single word without attempting to be parsed.

3.2 The Random Word Detection Module (RWDM)

In URLs used in phishing attacks, words formed from random characters can be used. A project in Github [14] was used to identify random words. The Markov Chain Model was used in the study. First of all, the system is trained in a given English text. The probability of finding two successive consecutive characters in the training phase is calculated. Characters can only be letters or space characters found in the dictionary. The probabilities of the characters other than these are not extracted.

In order to be able to understand whether a word is random, consecutive letter in the word extracted during the test phase. For the extracted letter pairs, the probabilities of the letter pairs learned during training are found and the probabilities for all letter pairs are multiplied. If the analyzed word looks like a real word, then the multiplicative result is a high value, otherwise the multiplicative result is a low value.

A threshold value is determined so that it can be decided that a word is random. If the multiplication of the likelihoods of the letter pairs is below this threshold value, the word is classified as random.

3.3 Maliciousness Analysis Module (MAM)

In this study, MAM has been developed to detect whether the words used in phishing attacks are used for fraudulent purposes or not. In this module words written with Typosqautting can be detected.

The words to be analyzed are analyzed by giving this module. The flow diagram showing the operation of this module is given in Fig. 6 [16].

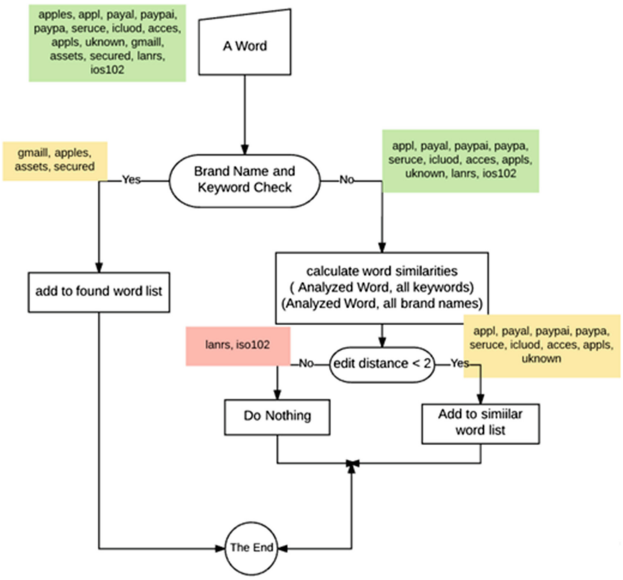


Fig. 6. MAM flow diagram

The words gathered from WDM may include a brand name or keyword. Therefore, the words to be analyzed are checked by brand name and keyword lists in this module again. Then the similarity is calculated by comparing the analyzed word with all the words in the brand name and keyword lists. Levenshtein Distance (Edit Distance) algorithm is used in the calculation of similarity. Then the similarity is calculated by comparing the analyzed word with all the words in the brand name and keyword lists. Levenshtein Distance (Edit Distance) algorithm is used in the calculation of similarity.

The Levenshtein Distance algorithm calculates how many moves are needed to convert an analyzed word to target word. The number of moves required for the conversion process gives the distance of the two words. Transactions used for conversion are; adding characters, deleting characters, and changing characters.

4 Experimental Results

The tests carried out in this study are grouped under 3 main titles.

- *Testing with Natural Language Processing (NLP) Features:* Tests with NLP-based features determined after the data preprocessing steps described in Sect. 4.
- *Test with Word Vectors:* Tests made with the features obtained after the vectorization of the words in the URL.

- *Hybrid Tests:* Tests where the features obtained with the word vectors and the NLP features are used together.

Within the scope of the tests performed, 3 different algorithms have been tried. Algorithms tested; Random Forest (RF) is a tree based algorithm, Sequential Minimal Optimization (SMO) is a kernel based algorithm, and Naive Bayes (NB) is a statistical based algorithm.

During the tests, the data set collected as described in Sect. 3. In the collected data set, there are 73,575 URLs including 37,175 malicious URLs and 36,400 legal URLs. Tests processed on a MacBook Pro device with 8 GB of 1867 MHz DDR3 RAM and 2.7 GHz Intel Core i5 processor. Weka [15] was used for testing. 10-fold Cross Validation and the default parameter values of all algorithms were used during the tests.

Tests were performed on a 10% sub-sample set instead of testing all samples because of lack of test device capacity. 3,717 malicious URLs and 3.640 legal URLs was ware used during the tests.

4.1 Testing with NLP Features

Data preprocessing steps which are explained in Sect. 4 has facilitated the extraction of some distinctive features. In this section, the extracted features and the obtained success rates are evaluated.

In the implemented system, firstly a URL was parsed according to some special characters such as ("=", ".", "/", "&", "?"). The obtained raw word list was processed to extract distinguishing features for malicious URL detection. The number of extracted distinguished features is 40. Detailed information on these features is given in Table 1.

Table 1. NLP features

The name of feature			
Raw word count (1)	www, com (2)	Alexa check (2)	Known TLD (1)
Punny code (1)	Separated word count (1)	Target keyword count (1)	Average adjacent word length (1)
Average word length (1)	Keyword count (1)	Other words count (1)	Brand check for domain (1)
Longest word length (1)	Brand name count (1)	Digit count (3)	Target brand name count (1)
Shortest word length (1)	Similar keyword count (1)	Subdomain count (1)	Consecutive character repeat (1)
Standard deviation (1)	Special character (8)	Random domain (1)	Similar brand name count (1)
Adjacent word count (1)	Random word count (1)	Length (3)	

4.2 Test with Word Vectors

Word Vectorization is one of the main approaches which are used for machine learning based text processing. In this approach, the words found in the text are used as features instead of the features that are manually extracted in the text based data.

For each URL, the words obtained after applying the preprocessing step given in Fig. 3 have been converted into vectors. The *StringToWordVector* module in Weka is used for the vectorization process. The generated word vectors are processed by machine learning algorithms.

For the 7,357 URLs used for the test, 10,572 features were extracted after the vectorization process. Then Feature Selection was made to reduce the number of features. The CfsSubsetEval algorithm is used for feature selection. The algorithm has been run Forward with the BestFirst search method. With feature selection algorithm execution, the number of features has dropped from 10,572 to 238.

4.3 Hybrid Test

The Word Vectorization and Feature selection steps were repeated for Hybrid Tests. After the implementation of the feature selection step, 238 word features are obtained. Then 40 NLP features were added to the features obtained. Classification algorithms have been tested with a total of 278 features.

4.4 Test Results

40 features are used for Testing NLP Features, 238 features are used for Testing Word Vectors and 278 features are used for Hybrid Tests. The F-measurement values for the achievement values obtained as a result of the tests performed are given in Table 2.

Table 2. Test Results

Algorithm	NLP features	Word vector	Hybrid
Random Forest	0.966	0.868	0.972
SMO	0.941	0.865	0.964
Naïve Bayes	0.655	0.817	0.755

According to the experimental results given in Table 2, it is seen that the system formed by the NLP features is more successful than the system formed by the Word Vector features. Hybrid approach using NLP Features in conjunction with Word Vector Features has been observed to have a boosting effect for Random Forest and Sequential Minimal Optimization algorithms.

In the previous work [16] of this study, we constructed our system with 17 NLP based features and 209 Word Vector features (and totally 226 features). The enhancement in this study increased the success of our prediction about 7%.

5 Conclusion

In this paper, a system has been developed to detect URLs which are used in Phishing Attacks. In the proposed system some features have been taken out by using NLP techniques. The extracted features are evaluated in two different groups. The first one is a person-determined attribute that is thought to be distinctive to malicious URLs and legal URLs. The second group focuses on the usage of the words in the URL without performing any other operations by applying only the vectorization process. Experimental study is constructed over three different test scenarios, including tests for NLP-based features, tests for Word Vectors, and Hybrid approach tests for both of these features.

During the tests; Random Forest which is a tree based algorithm, Sequential Minimal Optimization which is a kernel based algorithm and Naive Bayes algorithm which is a statistical based algorithm are used. According to the results obtained, the tests made for the hybrid approach were more successful than the other tests. In the hybrid approach, the Random Forest Algorithm was observed to be more successful than the other algorithms tested with 97.2% success rate.

Acknowledgement. Thanks to Normshield Inc., BGA Security, SinaraLabs and Roksit for contributing to the development of this work.

References

1. Anti-Phishing Working Group (APWG): Phishing activity trends report—last quarter (2016). http://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf
2. Khonji, M., Iraqi, Y., Jones, A.: Phishing detection: a literature survey. IEEE Commun. Surv. Tutor. **15**(4), 2091–2121 (2013)
3. Garera, S., Provos, N., Chew, M., Rubin, A.D.: A framework for detection and measurement of phishing attacks. In: Proceedings of the 2007 ACM Workshop on Recurring Malcode, pp. 1–8. ACM, November 2007
4. Stone, A.: Natural-language processing for intrusion detection. Computer **40**(12), 103–105 (2007)
5. Fu, A.Y., Wenyin, L., Deng, X.: Detecting phishing web pages with visual similarity assessment based on earth mover's distance (EMD). IEEE Trans. Dependable Secur. Comput. **3**(4), 301–311 (2006)
6. Toolan, F., Carthy, J.: Phishing detection using classifier ensembles. In: 2009 eCrime Researchers Summit, eCRIME 2009, pp. 1–9 (2009)
7. Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S.: A comparison of machine learning techniques for phishing detection. In: Proceedings of the Anti-Phishing Working Groups 2nd Annual eCrime Researchers Summit, eCrime 2007, pp. 60–69. ACM, New York (2007)
8. Cook, D.L., Gurbani, V.K., Daniluk, M.: Phishwish: a stateless phishing filter using minimal rules. In: Financial Cryptography and Data Security, pp. 182–186. Springer (2008)
9. Cao, Y., Han, W., Le, Y.: Anti-phishing based on automated individual white-list. In: DIM 2008: 4th ACM Workshop on Digital Identity Management, New York, pp. 51–60 (2008)

10. Sahingoz, O.K., Erdogan, N.: RUBDES: a rule based distributed event system. In: 18th International Symposium on Computer and Information Sciences - ISCIS 2003, Antalya, Turkey, pp. 284–291 (2003)
11. Phistank: join the fight against phishing. https://www.phishtank.com/developer_info.php. Accessed Oct 2017
12. Yandex account: Yandex Technologies. https://tech.yandex.com.tr/xml/. Accessed Oct 2017
13. PyEnchant—PyEnchant v1.6.6 documentation. http://pyenchant.readthedocs.io/en/latest/index.html. Accessed Oct 2017
14. A small program to detect gibberish using a Markov chain. https://github.com/rrenaud/Gibberish-Detector. Accessed Oct 2017
15. Weka 3: data mining software in Java. http://www.cs.waikato.ac.nz/ml/weka/. Accessed Oct 2017
16. Buber, E., Diri, B., Sahingoz, O.K.: Detecting phishing attacks from URL by using NLP techniques. In: 2017 International Conference on Computer Science and Engineering (UBMK), Antalya, Turkey, pp. 337–342 (2017)