Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 129 (2018) 224–230

Procedia Computer Science

www.elsevier.com/locate/procedia

2017 International Conference on Identification, Information and Knowledge in the Internet of Things

Detecting Phishing Websites via Aggregation Analysis of Page Layouts

Jian Mao[a,*], Jingdong Bian[a], Wenqian Tian[a,b], Shishi Zhu[a], Tao Wei[c], Aili Li[d], Zhenkai Liang[e]

[a]*School of Electronic and Information Engineering, Beihang University, Beijing 100183, China*
[b]*Shanghai Key Laboratory of Integrated Administration Technologies for Information Security, Shanghai 200240, China*
[c]*Baidu USA LLC., Sunnyvale, CA 94089, USA*
[d]*Information Technology Service Center, China National Petroleum Corporation, Beijing 100007, China*
[e]*School of Computing, National University of Singapore, Singapore 117417, Singapore*

Abstract

Phishing websites are typical starting points of online social engineering attacks, including many recent online scams. The attackers develop web pages mimicking legitimate websites, and send the malicious URLs to victims to lure them to input their sensitive information. Existing phishing defense mechanisms are not sufficient to detect with new phishing attacks. In this paper, we aim to improve phishing detection techniques using machine learning techniques. In particular, we propose a learning-based aggregation analysis mechanism to decide page layout similarity, which is used to detect phishing pages. Our experiment results shows that our approach is accurate and effective in detecting phishing pages.

Keywords: Anti-phishing; Web security; Machine learning; Aggregation analysis

1. Introduction

Phishing websites are typical entrance points of online social engineering attacks, including many recent online scams. In such attacks, the attackers develop web pages mimicking legitimate websites, and send the malicious URLs to victims via spam emails, instant messages, or social network communications. Their goal is to deceive the victims to input their sensitive information (e.g., bank accounts, social security number, etc.). Though phishing attacks do not require advanced technical knowledge and these attack techniques are becoming familiar to users, they are still causing major financial damages. These attacks also negatively influence users' trust toward the web services greatly.

* Jian Mao. Tel.: +86-10-8231-7212; Fax: +86-10-8231-9474.
E-mail address: maojian@buaa.edu.cn

10.1016/j.procs.2018.03.053

According to the report from anti-phishing working group (APWG), there are 1,220,523 phishing attacks reported in 2016, which is a 65% increase over 2015 [3].

Several types of anti-phishing solutions have been developed. The traditional URL-based antiphishing solutions [18, 9, 20, 12, 11] are limited by the timeliness of malicious URL database update. The solutions based on page contents [26, 17] heavily rely on the context or image processing techniques, which cause high performance overhead. As the phishing pages usually maintain similar page layouts to their target websites, the similarity of page layouts has been demonstrated as an important metric to detect phishing pages [14]. However, these metrics are derived from human experiences, and thus may not be comprehensive to detect new attacks. How to comprehensively evaluate the pages' similarity remains a great challenge.

In this paper, we explore learning techniques to address this problem. Our solution is based on the aggregation analysis mechanism to automatically generate rules to determine layout similarity of web pages and then detect phishing pages. It first trains a similarity classifier using page layout features, then uses the classifier to detect phishing pages. Our evaluation used more than 2,900 phishing web pages from *phishtank.com*. It shows that our approach is effective in creating classifiers and detecting phishing pages via page layout similarity.

In summary, we made the following contributions in this paper:

- We propose a learning-based mechanism to evaluate the similarity of web page layouts and identify phishing pages.
- We define the rules to extract and create effective page layout features and develop a phishing page classifier based on two typical learning algorithms, supporting vector machine and decision tree.
- We prototyped our approach and evaluated it with real-world web page samples from *phishtank.com* and *alexa.com*.

Paper Organization. The rest of this paper is organized as follows. Section 2 introduce the background of our work and gives an overview of our approach. Section 3 presents our main algorithm. Section 4 presents the evaluation results. We discuss closely related work in Section 5 and conclude the paper in Section 6.

2. Overview

In this section, we introduce the problem faced by the layout-based similarity detection and give an overview of our solution.

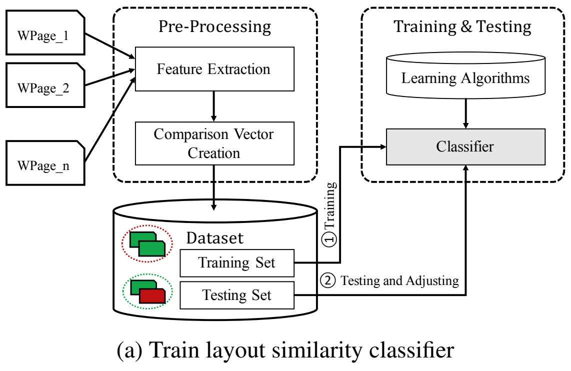
2.1. Learning-based Layout Similarity Detection

Cascading Style Sheets (CSS) is the commonly used visual layout definition of web pages. Widely support by browsers, CSS rules specify how different classes of web page components should appear, for example, the font type and color of the body of a page.

In our previous work [14], we have demonstrated that CSS-based page layouts can be used as the basis to detect phishing pages. However, the metric used is mainly based on human experiences, and may not comprehensively represent all the statistical similarity properties between page layouts of phishing pages and legitimate pages. Especially, the *threshold*, a critical parameter of that approach is selected based on the similarity score distribution of the collected samples. As a result, its accuracy heavily relies on the completeness of the sample collection and attackers may craft new phishing pages to bypass the detection.

Our goal is to develop methods that can detect the similarity among two page layouts by comprehensively "considering" layout features. Machine learning mechanisms are typically used in such situations, where they are used to infer similarity models according to the statistical properties retrieved from the training samples.

In this work, we integrate and analyze a few of potential learning algorithms. Support Vector Machine (SVM) [23] is a widely used classification algorithm due to its good performance. The basic idea of SVM is to maximize the margin between two classes closest points and find an optimal separating hyperplane between them. Decision Tree (DT) learning [22] is one of the predictive modelling algorithms. It takes a decision tree as the predictive model and



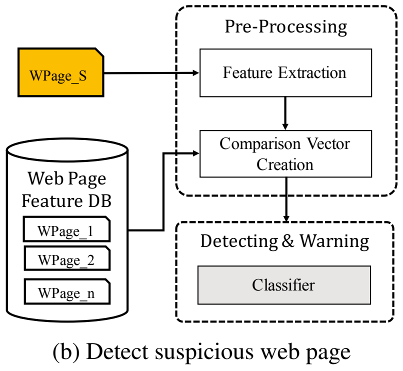


Fig. 1: Overview of our approach.

determines an item's target value (represented as a leaf) according to the observations about the item (represented in branches).

2.2. *Approach Overview*

The problem addressed by our paper can be formulated as follows: Taking a suspicious page and a set of benign pages as inputs, we aim to extract features and apply learning algorithms to detect phishing pages comprehensively based on its layout similarity features.

As shown in Fig. 1, our approach includes two phases: *similarity classifier training* and *phishing web page detection based on layout similarity*.

2.2.1. *Similarity classifier training*

We first obtain a classifier to decide page similarity from layout features. This phase consists of the *pre-processing* stage and the *training* stage. The pre-processing stage takes as inputs two categories of pre-prepared web page pairs, visually similar web page pairs and visually different web page pairs. Our approach obtains features from web page layouts and creates the *comparison vectors*, which summarizes the key similarity features, of every web page pairs accordingly. We labelled the comparison vector as "c_1' to represent a pair of similar web pages, correspondingly, the visually different web page pair is labelled as "c_0". We split the comparison vectors obtained into the *training* set and the *testing* set.

The classifier training stage takes as inputs the labelled comparison vectors from the training set. The similar page classifier obtained in this stage can be used to determine whether two web pages are similar according to their comparison vector.

2.2.2. *Phishing web page detection based on layout similarity*

The trained classifier can then be used to detect phishing pages. When a user opens a new web page, "WPage_S" (illustrated in a yellow block in Fig. 1(b)), our detector pre-processes the web page by extracting the layout features of the new page and creating comparison vectors between the "WPage_S" and the pages, "WPage_1, ..., WPage_n" in web page database *Web Page Feature DB*, respectively. The classifier obtained in Phase I takes the comparison vectors as inputs and determines the labels of each vector. If the comparison vector of web page pair "(WPage_S, WPage_i)" is classified as "c_0", it means "WPage_S" is visually different from "WPage_i" and the system will go to test the next vector. Otherwise, it means "WPage_S" is visually similar as "WPage_i". One the classifier outputs a "c_1" labelled vector, the system will send a warning message to alert users.

3. Learning-based Similar Page Layout Classification

In this section, we introduce our learning-based page-layout similarity classification mechanism in detail. In our classifier, we take labelled comparison vectors as inputs (where 1 denotes that the two pages are similar and 0 denotes that the two pages are not similar). The output of our classifier is 1 (similar) or 0 (different).

3.1. Property Vector Extraction

In this step, we present the rules that quantify the CSS elements' impact of a web page and combine CSS features of two pages into one comparison property vector.

3.1.1. Property Vector Generation

As in our previous work [14], we use the area of elements in a page to demonstrate its impact on page layout. The larger the area is, the more impact it has on the page layout. In this paper, we use the relative area, i.e., the proportion of an element's area to by the whole page window size, to avoid the inaccuracy of detection in different page window sizes.

As page visual appearance is affected by CSS selectors' properties and values not CSS names of selectors, we extract and express CSS features to the pattern denoted as follow:

```
Property1 {Value1-1: Area proportion of Selector1-1; Value1-2: Area proportion of Selector1-2; ...};
Property2 {Value2-1: Area proportion of Selector2-1; Value2-2: Area proportion of Selector2-2; ...};
...
```

In this pattern, we rank the CSS objects in the decreasing order by area proportion, i.e., $Area\ proportion\ of\ Selector_1$-1 $\geq Area\ proportion\ of\ Selector_1$-2 $\geq ... \geq Area\ proportion\ of\ Selector_1$-$n$.

However, different pages have different numbers of CSS selectors and declarations. If we want to merge two pages, we should unify the dimension of properties of different pages and then they can be combined. We make a union set of the properties all the web pages collected for training and testing, denoted by Σ, where $\Sigma = \{Property_1, Property_2, ..., Property_k\}$. We make the length of the union set $|\Sigma| = k$ as the dimension of the property vector. So, we can unify effective CSS features of one page into the following pattern:

```
Page1 [Property1{..}, Property2{..},..., Propertyk{..}]
```

Different pages have the same k property expressed as above. For simplicity, we write the pattern as follows:

```
Page1 [P1{V1-1:A1-1,...,V1-m1:A1-m1}, P2{V2-1:A2-1,...,V2-m2:A2-m2} ,...,
Pk{Vk-1:Ak-1,...,Vk-mk:Ak-mk}]
```

where P denotes the *Property*, V denotes the *Value*, A denotes the *Area proportion*. Different *Properties* may have different numbers of *Values*.

3.1.2. Comparison Vector Generation

Given two pages, we quantify their common CSS features into a comparison vector. The procedure is as follows.

- In the same property P_k, Page1 has m_1 Values $\{V_k$-1:A_k-1,...,V_k-m_1:A_k-$m_1\}$ and Page2 has m_2 Values $\{V_k$-1:A_k-1,...,V_k-m_2:A_k-$m_2\}$. Choose the maximum of m_1 and m_2 denoted by m and extend the smaller one to the length of m by adding zeros. The outputs in this step are Page1 $\{V_k$-1:A_k-1,...,V_k-m:A_k-$m\}$ and Page2 $\{V_k$-1:A_k-1,...,V_k-m:A_k-$m\}$ with the same dimension.
- Compute the difference between Page1:V_k-i and Page2:V_k-i where $i \in m$ and use the maximum value of Page1:A_k-i and Page2:A_k-i to multiply the difference value. The result is denoted by ε_k-i.
  ε_k-i = |Page1:V_k-1 - Page2:V_k-1| × max(Page1:A_k-i,Page2:A_k-i).
  Then we get a value in i_{th} $\{V_k$:$A_k\}$ as the i_{th} dimension of their comparison property vector.
- Calculate all the ε_k-i of P_k and obtain ε_k=sum(ε_k-i where $i = 1, ..., m$).
- After repeating the previous steps k times, we finally get the comparison property vector of Page1 and Page2 denoted as $[\varepsilon_1, \varepsilon_2, ... ,\varepsilon_k]$.

3.2. Classifier Building

We consider our approach as a 2-category classification problem. We set the output of the classifier as a binary output, 1 or 0, and make the comparison property vectors in the dataset as inputs. We divide the dataset into two parts. One is used to train the classifier and the other as testing set is used to optimize the performance. Let $\Gamma_1 = \{x_i\}_{i=1}^{m}$ be a set of M training vectors, where x_i is a k-dimension vector labelled by $y_i \in \{\pm 1\}$, with $y_i = 1$ and $y_i = -1$ indicating x_i to the Class 1 and Class 2 respectively. And $\Gamma_2 = \{x_i\}_{i=1}^{n}$ be a set of N testing vectors.

We employ two classic classifiers in our approach respectively, including Support Vector Machine (SVM) [7], DecisionTree (DT) [5]. We use Γ_1 to train a classifier model and use Γ_2 to test its performance. When the input of a comparison property vector gets output 1, which means the two pages are similar. The suspicious page will be determined to be malicious. When the input of a comparison property vector gets output 0, which means the two pages are not similar. The suspicious page will be determined to be benign. We evaluate four classifiers in the next section.

4. Evaluation

We deploy two machine learning classifiers to evaluate the performance of our solution. Especially, we use four metrics, *accuracy, precision, recall, and F1 score*, to analyze the results.

Dataset Preparation.. We collected phishing websites from *phishtank.com*. We first checked and filtered those invalid pages manually. We then excluded those pages whose layout elements are too small and whose layout appearance is totally different from their target. We selected 13 target pages and 102 suspicious pages to test our approach. In *Property Vector Extraction*, we obtained 1,400 comparison vectors as positive samples whose label are set to 1 (1200 samples for training) and 1,523 comparison vectors as negative samples whose label are set to 0 (1200 samples for training). There were 2,923 samples in total to train and test our two classifiers.

Support Vector Machine(SVM).. We employ Support Vector Machine(SVM) as the classifier and we test four metrics regarding to the parameter *gamma* in SVM algorithm. According to the experiment results shown in Figure 2 (a), all of the four metrics are beyond 90% and with the rising of *gamma*, the value of precision rises a little while recall falls down a little. When *gamma* is about 0.00005, the four metrics get close to the best performances.

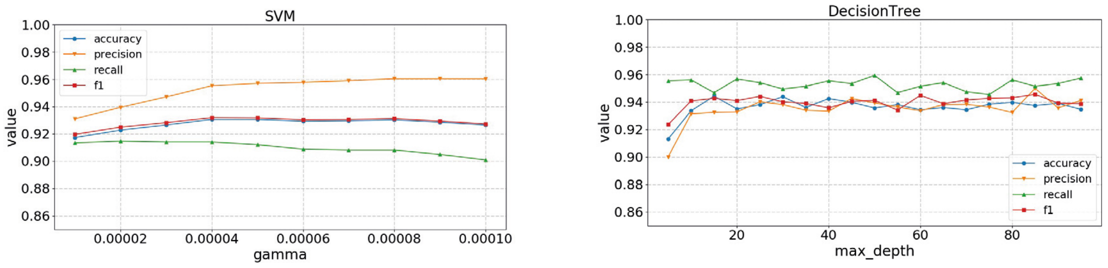


Fig. 2: Evaluation Results. (a) Result of SVM with regard to *gamma*; (b) Result of DecisionTree with regard to *depth of tree*

DecisionTree (DT).. We employ DecisionTree(DT) as the classifier and we test four metrics regarding to the parameter *max_depth* in DT algorithm. The results are shown in Figure 2 (b), where the four metrics are beyond 90% and their values may fluctuate a little. When *max_depth* is about 85, the four metrics achieve the best. According the experiment results, two classifiers both have more than 93% accuracy and more than 95% precision, which demonstrates that our approach can make an effective detection in phishing websites.

5. Related work

In this section, we discuss the work from past research that is closely related to our approach.

Page-feature based solutions.. Eric et al. [15] proposed a scheme that selects text pieces, images and overall visual appearance as the basic properties to compare the similarity of two pages. Chen et al. [6] presented another visually similar web page detection algorithm according to Getstalt theory, in which they process the webpage as an indivisible entity. CANTINA [26] detects phishing pages based on “term frequency-inverse document frequency (TF-IDF)”. SpoofGuard [8] uses *domain name*, *URL*, *link* and *image* as the critical features to check suspicious pages. GoldPhish [10] uses optical character recognition from a rendered page to extract page information. It then uses search engines to decide whether the page content is consistent with its domain, and thus identifies phishing sites. Zhang et al. [25] uses spatial layout characteristics from web pages and uses as a basis to decide page similarity. Moghimi et al. [16] discovered a rule-based scheme used two novel feature sets to detect phishing in internet banking. One feature set is used to evaluate the identity of page resources and the other is utilized to identify the access protocol. Wardman et al. [21] uses file-level similarity between two web pages and to detect phishing web sites. Phishing-Alarm uses CSS layout features are efficient and robust in detecting phishing web sites [14]. In contrast to identify new features as a basis for phishing detection, this paper focuses on how to automatically learn classifiers of similar pages from CSS features.

Learning based solutions.. Machine learning has been applied to web page classification in detecting phishing. Pan et al. [19] presented an SVM-based page classifier for detection of phishing sites. Xiang et al. [24] proposed *CANTINA+* that takes the 15 features from URL, HTML DOM (Document object model), third party services, and search engines. It trains these features using SVM (Support vector machine) to detect phishing attacks. Abu-Nimeh et al. [2] compared six machine learning algorithms for phishing detection, including Bayesian Additive Regression Trees, Logical Regression, SVM, RF, Neural Network, and Regression Tree. Lee et al.[13] leverages a linear chain CRF model to understand web browsing behaviors of users on phishing web sites. It then predicts behavior under the context to detect phishing attacks. Abdelhamid et al. [1] proposed an associative classification method for web site phishing detection based on multi-label classifiers. Bottazzi et al. [4] proposed a framework in Android mobile devices for phishing detection, which includes a machine learning detection engine for protecting from new phishing activities.

6. Conclusion

In phishing web site detection, comprehensively evaluating page similarity remains a great challenge. In this paper, we propose a learning-based aggregation analysis mechanism to determine similarity of page layouts and detect phishing pages. Our approach automatically trains classifiers to determine web page similarity from CSS layout features, which does not require human expertise. We prototyped our approach and evaluated it using a large amount of phishing web pages. The experiment results demonstrate that our approach is accurate and effective in determining similarity from page layouts. Our approach can effectively enhance the performance of existing antiphishing mechanisms.

Acknowledgements

We thank the anonymous reviewers for their valuable comments. This work was supported in part by the National Key R&D Program of China (No. 2017YFB0802400), the National Natural Science Foundation of China (No. 61402029), the Funding Project of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security (No. AGK201708).

References

[1] Abdelhamid, N., Ayesh, A., Thabtah, F., 2014. Phishing detection based associative classification data mining. Expert Systems with Applications 41 (13), 5948–5959.

[2] Abu-Nimeh, S., Nappa, D., Wang, X., Nair, S., 2007. A comparison of machine learning techniques for phishing detection. In: Anti-Phishing Working Groups Ecrime Researchers Summit. pp. 60–69.

[3] APWG, 2016. Statistical highlights for 4th quarter 2016. http://www.http://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf.

[4] Bottazzi, G., Casalicchio, E., Cingolani, D., Marturana, F., Piu, M., 2015. MP-shield: A framework for phishing detection in mobile devices. Proceedings - 15th IEEE International Conference on Computer and Information Technology, CIT 2015, 14th IEEE International Conference on Ubiquitous Computing and Communications, IUCC 2015, 13th IEEE International Conference on Dependable, Autonomic and Se, 1977–1983.

[5] Breiman, L. I., Friedman, J. H., Olshen, R. A., Stone, C. J., 1984. Classification and regression trees (cart). Biometrics 40 (3), 358.

[6] Chen, T.-C., Dick, S., Miller, J., May 2010. Detecting visually similar web pages: Application to phishing detection. ACM Transaction on Internet Technology 10 (2), 1–38.

[7] Cherkassky, V., 1997. The nature of statistical learning theory . IEEE Transactions on Neural Networks 8 (6), 1564.

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

[9] C.Inc., August 2015. Couldmark toolbar. http://www.cloudmark.com/desktop/ie-toolbar.

[10] Dunlop, M., Groat, S., Shelly, D., May 2010. Goldphish: Using images for content-based phishing analysis. In: Internet Monitoring and Protection (ICIMP), 5th International Conference on. IEEE, Barcelona, pp. 123–128.

[11] Fette, I., Sadeh, N., Tomasic, A., May 2007. Learning to detect phishing emails. In: Proceedings of the International World Wide Web Conference (WWW).

[12] iTrustPage, 2013. http://www.cs.toronto.edu/ronda/itrustpage/.

[13] Lee, L.-h., Lee, K.-c., Juan, Y.-c., Chen, H.-h., Tseng, Y.-h., 2014. Users ' Behavioral Prediction for Phishing Detection. In: Proceedings of the 23rd International Conference on World Wide Web. No. 1. pp. 337–338.

[14] Mao, J., Tian, W., Li, P., Wei, T., Liang, Z., 2017. Phishing website detection based on effective css features of web pages. In: The 12th International Conference on Wireless Algorithms, Systems, and Applications. pp. 804–815.

[15] Medvet, E., Kirda, E., Kruegel, C., September 2008. Visual-similarity-based phishing detection. In: Proceedings of SecureComm 2008. ACM.

[16] Moghimi, M., Varjani, A. Y., 2016. New rule-based phishing detection method. Expert Systems with Applications 53, 231–242.

[17] Nourian, A., Ishtiaq, S., Maheswaran, M., 2009. Castle: A scocial framework for collaborative anti-phishing databases. ACM Transactions on Internet Technology.

[18] P., L., Jung, E., D., D., T.E., H., J.P., H., May 2008. B-apt: Bayesian anti-phishing toolbar. In: Proceedings of IEEE International Conference on Communications, ICC'08. IEEE Press.

[19] Pan, Y., Ding, X., 2006. Anomaly based web phishing page detection. In: Computer Security Applications Conference, 2006. ACSAC '06. pp. 381–392.

[20] Ronda, T., Saroiu, S., Wolman, A., April 2008. itrustpage: A user-assisted anti-phishing tool. In: Proceedings of Eurosys'08. ACM.

[21] Wardman, B., Stallings, T., Warner, G., Skjellum, A., Nov 2011. High-performance content-based phishing attack detection. In: eCrime Researchers Summit. IEEE, San Diego, CA, pp. 1–9.

[22] Wikipedia, 2017. Decision tree learning. https://en.wikipedia.org/wiki/Decision_tree_learning/, [Online].

[23] Wikipedia, 2017. Support vector machine. https://en.wikipedia.org/wiki/Support_vector_machine/, [Online].

[24] Xiang, G., Hong, J., Rose, C. P., Cranor, L., sep 2011. CANTINA + : A Feature-rich Machine Learning Framework for Detecting Phishing Web Sites. ACM Transactions on Information and System Security (TISSEC) 14 (2), 21.
URL http://dl.acm.org/citation.cfm?doid=2019599.2019606

[25] Zhang, W., Lu, H., Xu, B., Yang, H., 2013. Web phishing detection based on page spatial layout similarity. Informatica 37 (3), 231–244.

[26] Zhang, Y., Hong, J., Cranor, L., May 2007. Cantina: A content-based approach to detecting phishing web sites. In: Proceedings of the International World Wide Web Conference (WWW).