Automatic Detection of Phishing Target from Phishing Webpage

Gang Liu, Bite Qiu, Liu Wenyin
Department of Computer Science, City University of Hong Kong,
83 Tat Chee Ave., HKSAR, China
gangliu@student.cityu.edu.hk, biteqiu@cityu.edu.hk, csliuwy@cityu.edu.hk

Abstract

An approach to identification of the phishing target of a given (suspicious) webpage is proposed by clustering the webpage set consisting of its all associated webpages and the given webpage itself. We first find its associated webpages, and then explore their relationships to the given webpage as their features for clustering. Such relationships include link relationship, ranking relationship, text similarity, and webpage layout similarity relationship. A DBSCAN clustering method is employed to find if there is a cluster around the given webpage. If such cluster exists, we claim the given webpage is a phishing webpage and then find its phishing target (i.e., the legitimate webpage it is attacking) from this cluster. Otherwise, we identify it as a legitimate webpage. Our test dataset consists of 8745 phishing pages (targeting at 76 well-known websites) selected from PhishTank and preliminary experiments show that the approach can successfully identify 91.44% of their phishing targets. Another dataset of 1000 legitimate webpages is collected to test our method’s false alarm rate, which is 3.40%.

1. Introduction

Phishing is a kind of online attack widely used by phishers to steal users’ accounts and passwords, and other personal information for illegal appropriation. In recent years, phishing attacks have become more and more sophisticated and their volume expands dramatically. According to the Anti-phishing Working Group [1], 211,271 unique phishing websites were reported in the first half of 2009 and about 279 reputable brands were hijacked every month.

In order to direct users to fraudulent web sites and steal their money, phishing patterns evolve constantly by phishers. However, the essence of phishing has no change. Generally, most phishing webpages use links pointed to legitimate webpages and visually similar content to lure visitors to enter their sensitive information. In this sense, phishing webpages are not isolated from their targets but have strong relationships with them, which can be used as clues to find their targets.

In this paper, we propose an approach to automatic identification of the phishing target of a given webpage by clustering the webpage set consisting of all its associated webpages and the given webpage itself. The associated webpages are those which are pointed by forward links of the given webpage and webpages returned by a powerful search engine with certain representative keywords (e.g. brand, title and keywords of content) in the given webpage as queries. This approach first finds its associated webpages and then mines the features such as links relationship, ranking relationship, webpage text similarity and webpage layout similarity relationship between the given webpage and its associated webpages. Finally, a DBSCAN [2] clustering method is employed to find if there is a cluster around the given webpage. If such cluster is found, the given webpage is regarded as a phishing webpage and its target is identified as the legitimate webpage in the cluster which is closest to the given webpage. Otherwise, we identify it as a legitimate webpage.

2. Related Work

Existing anti-phishing methods can be mainly classified into the following categories.

* The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 117907].
3.1. Finding the associated webpage set

The associated webpages are mainly from two sources.

(1) Directly associated pages. They are the webpages that are directly linked by the given webpage \( P \). They can be found by examining the HTML source of page \( P \) and extracting all hyperlinks in it.

(2) Indirectly associated pages. They are the pages which share the same or similar text/visual information with \( P \). For example, we can mine such indirectly associated pages of \( P \) by searching the Web with the representative keywords found in \( P \) as queries. The representative keywords are those found in the title, words in Meta tag, keywords in the body tag and organization name of \( P \).

3.2. Representing webpages in feature vectors

The strength values of the association relationships are defined and measured as follows.

**Feature 1: link relationship**

We use \( L_{ij} \) as the metrics for the link relationship from page \( i \) to page \( j \).

\[
L_{ij} = \frac{NL_{ij}}{NL_i},
\]

where, \( NL_{ij} \) is the number of forward links from page \( i \) to any page on the website of page \( j \); \( NL_i \) is the total number of forward links from page \( i \).

**Feature 2: ranking relationship**

We define the ranking association relationship from page \( i \) to page \( j \) based on the rank of page \( i \) in the search result using the representative keywords of page \( j \) as the query. The strength (degree) of the ranking relationship is measured by the rank of the domain name of page \( j \) in the search results. \( R_{ij} \) is used as the metrics for ranking relationship from page \( i \) to page \( j \).

\[
R_{ij} = \frac{N_j - (R_i - 1)}{N_j},
\]

where \( N_j \) is the total number of search results with which we are concerned and \( R_i \) is the rank of the domain name of page \( i \) in the results. If it cannot be found in the result, its rank value is set as zero.

**Feature 3: text similarity relationship**

We measure text similarity relationship \( TS_{ij} \) from page \( i \) to page \( j \) as follows.

\[
TS_{ij} = \cos(x, y) = \frac{x \cdot y}{\|x\| \|y\|},
\]

where, \( x \) is the term vector extracted from page \( i \); \( y \) is the term vector extracted from page \( j \); \( \|x\| \) is the length of vector \( x \), \( \|y\| \) is the length of vector \( y \).

**Feature 4: layout similarity relationship**

We define the layout similarity \( LS_{ij} \) from page \( i \) to page \( j \) as the ratio of the weighted number of matched blocks to the total number of blocks in page \( j \). Two blocks are considered matched if they both exhibit high visual similarity and satisfy the same constraints with corresponding matched blocks.
where, $T_i(blocks)$ and $T_j(blocks)$ denote the blocks set of page $i$ and page $j$, respectively; $|T_i(blocks)|$ is the total number of blocks in page $i$; $|T_j(blocks) \cap T_i(blocks)|$ is the number of the blocks they share.

We quantify those association relationships from the given webpage $P$ to any page ($page$) in the associated webpage set and use them as features in association relationship with $P$. In addition, it can select any data point as the start point for clustering. We select the given webpage as the start point for our purpose. The closer to $P$, the greater the data point’s density is. Hence, we employ the well-known density-based DBSCAN algorithm. In addition, it can select any data point as the start point for clustering. We select the given webpage as the start point for our purpose. The concepts used in our approach are described as follows.

**Eps:** Maximum radius of the neighbourhood of the cluster.

**MinPts:** Minimum number of points in an Eps-neighbourhood of that point.

**core point (CO):** Point is in the interior of a density-based cluster.

**border point:** A border point is not a core point, but falls within the neighbourhood of a core point.

**directly-density-reachable (DDR):** If point $x$ is CO, point $y$ is in x’s Eps-neighbourhood.

**density-reachable:** There exists a chain of DDR objects from point $x$ to point $y$.

Based on the above concepts, we present our clustering-based method as follows:

1) Find the associated webpage set of $P$ and also add $P$ into the set.

2) Quantify the association relationships from $P$ to each webpage in the set and represent each webpage in a feature vector.

3) Select the given webpage $P$ as the start point and retrieve all points density-reachable from $P$ with respect to $Eps$ and $MinPts$.

4) If $P$ is a core point, a cluster is formed and the process stops.

5) If $P$ is a border point and no point is density-reachable from $P$, no cluster is formed and the process stops.

6) If $P$ is a border point and no point is density-reachable from $P$, no cluster is formed and the process stops.

If we can find a cluster including $P$, we claim that $P$ is a phishing webpage and the page which has the strongest association relationship with $P$ is its phishing target.

### 4. Experiments and Evaluation

We have implemented our method as a windows application accepting a URL as the given page. The user interface of the application is shown in Figure 1. Eps and MinPts are two parameters of the DBSCAN algorithms. If the given page is phishing, all the webpages it is potentially attacking are listed.

![Figure 1. Interface of our windows application: Automatic Identification of Phishing Target](image)

#### 4.1. Datasets and experiment results

In our experiment, we selected 8745 phishing URLs targeting at 76 well-known companies/brands from PhishTank [11] to test the phishing target identification accuracy of our method. In our experiment, an identified phishing target is considered as correct for the given webpage if their domain name matches. Our method’s accuracy rate of identification is 91.44%. Another testing dataset consists of 1000 legitimate pages which were randomly obtained from Random Yahoo Link [12]. The false alarm rate is 3.40%.

#### 4.2. Empirical parameters setting

In this experiment, we tune the two parameters, Eps and MinPts to find their optimal values such that we can achieve the maximum average accuracy rate in detecting phishing targets and minimum average false alarm rate. Figure 2 and figure 3 depict the results using different Eps and MinPts.

As the results show, when the value of MinPts increases, bigger clusters are more likely to be labeled as noise and the target has less possibility to be identified. Hence the accuracy rate of phishing target detection deceases and the false alarm rate is also decreased. As for the parameter Eps, when it increases,
it is more likely to form a cluster. Thus the accuracy rate of phishing target detection is improved and the false alarm rate should also be increased. After our observation on the real data from these two figures, we conclude that our method provides the best performance with $Eps = 0.11$ and $MinPts = 4$, where the accuracy rate is 91.44% and the false alarm rate is 3.40%.

![Figure 2. Accuracy rate with different $Eps$ and $MinPts$](image)

![Figure 3. False alarm rate with different $Eps$ and $MinPts$](image)

4.3. Comparison with SLN-based method

We compare our method with a SLN-based method [10] in terms of performance. The detailed description of the comparison is as follows: (1) phishing target detection accuracy and (2) false alarm rate test.

<table>
<thead>
<tr>
<th>Anti-phishing methods</th>
<th>Phishing target detection</th>
<th>False alarm rate test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accuracy rate</td>
<td># of Dataset</td>
</tr>
<tr>
<td>SLN method</td>
<td>83.40%</td>
<td>1000 pages</td>
</tr>
<tr>
<td>Our method</td>
<td>91.44%</td>
<td>8745 pages</td>
</tr>
</tbody>
</table>

Based on the comparison in Table 1, the accuracy of our method outperforms the SLN-based method by 8.04% for target detection while its false alarm rate is much lower than the SLN-based method.

5. Conclusion and Future Work

In this paper, we present a novel approach to identifying the potential phishing target of a given webpage. Our approach first mines its associated webpage set of the given webpage and systematically inspects those features of the link relation, ranking relation, similarity relation from the given webpage to its associated webpages. A DBSCAN clustering method is employed to find if there exists a cluster of webpages attacked by the given webpage potentially attacking. Experiments show that the accuracy rate is 91.44% and the false alarm rate is 3.40%.

In our future work, we plan to integrate this solution into web browsers as light-weight plug-ins to provide alerts for phishing attacks. We also plan to implement a class library (APIs) for enterprise users to build their own application system to check suspicious websites.

References