Phishing Detection based on Visual-Similarity

A.V.R.Mayuri., M.Tech, (Ph.D)

Abstract— Phishing is a form of online fraud that aims to steal a user's sensitive information, such as online banking passwords or credit card numbers. The victim is tricked into entering such information on a web page that is crafted by the attacker so that it mimics a legitimate page. Recent statistics about the increasing number of phishing attacks suggest that this security problem still deserves significant attention. In this paper, we present a novel technique to visually compare a suspected phishing page with the legitimate one. The goal is to determine whether the two pages are suspiciously similar. We identify and consider three page features that play a key role in making a phishing page look similar to a legitimate one. These features are text pieces and their style, images embedded in the page, and the overall visual appearance of the page as rendered by the browser. To verify the feasibility of our approach, we performed an experimental evaluation using a dataset composed of 41 real world phishing pages, along with their corresponding legitimate targets. Our experimental results are satisfactory in terms of false positives and false negatives.

Keywords - Anti-Phishing, Web document analysis, Security, Visual Similarity

1. INTRODUCTION

Phishing is a form of online fraudulent activity in which an attacker aims to steal a victim's sensitive information, such as an online banking password or a credit card number. Victims are tricked into providing such information by a combination of spoofing techniques and social engineering.

In practice, the victims often receive an email that tries to convince them to visit a web page that has been prepared by the attacker. This page mimics and spoofs a real service such as an online banking web site. Legitimately looking web forms are provided through which the attacker can harvest and collect confidential and sensitive information. Although tricking people to make financial profit is an old idea, criminals have realized that social-engineering-based attacks are simple to perform and highly effective over the Internet. Hence, although highly publicized, phishing is still an important security problem and many Internet users fall victim to this fraud. Note that such attacks are not only problematic for Internet users, but also for organizations that provide financial services online. The reason is that when users fall victim to phishers, the organization providing the online service often suffers an image loss as well as financial damage.

In recent years, phishing attacks have gained attention because their numbers and their sophistication have been increasing. The Anti Phishing Work Group detected more than 25,000 unique phishing URLs in December 2007 [1]. Also, creating a phishing site has become easier. The most straightforward and widespread method to commit a phishing attack, however, still consists of deploying a web page that looks and behaves like the one the user is familiar with. In this paper, we present an effective approach to detect phishing attempts by comparing the visual similarity between a suspected phishing page and the legitimate site that is spoofed. When the two pages are "too" similar, a phishing warning is raised. In our system, we consider three features to determine page similarity: text pieces (including their style-related features), images embedded in the page, and the overall visual appearance of the page as seen by the user (after the browser has rendered it). We quantify the similarity between the target and the legitimate page by comparing these features, computing a single similarity score. We chose to perform a comparison based on page features that are visually perceived. This is because phishing pages mimic the look-and-feel of a legitimate site and aim to convince the victims that the site they are visiting is the one they are familiar with. Once trust is established based on visual similarity, there is a higher chance that the victim will provide her confidential information. Typically, a victim's visual attention focuses both on the global appearance of the page and on salient details such as logos, buttons, and labels.

2. RELATED WORK

Phishing is an important security problem. Although phishing is not new and, hence, should be well-known by Internet users, many people are still tricked into providing their confidential information on dubious web pages. To counter the phishing threat, a number of anti-phishing solutions have been proposed, both by industry and academia. A class of anti-phishing approaches aims to solve the phishing problem at the email level. The key idea is that when a phishing email does not reach its victims, they cannot fall for the scam. This line of research is closely related to anti-spam research. One reason for the abundance of spam and phishing emails is the fact that the Simple Mail Transport Protocol (SMTP) does not contain any authentication mechanisms. The sender information in an email message can easily be spoofed. To address this problem, Microsoft and Yahoo have defined email authentication protocols (Sender ID [6]) that can be used to verify whether a received email is authentic. Unfortunately, however, these protocols are currently not employed by the majority of Internet users. Browser-integrated solutions to mitigate phishing attacks are SpoofGuard [3, 10] and PwdHash [9, 8]. SpoofGuard checks for phishing symptoms (such as obfuscated URLS) in web pages. PwdHash, in comparison, creates domain specific passwords that are rendered useless if they are submitted to another domain

3. OUR APPROACH

In the following subsection, we provide a high-level overview of how our system can be used to detect phishing pages. Then, we discuss how we extract signatures from web pages, and how we use these signatures to compare two pages to determine their visual similarity.

3.1 USING THE SYSTEM

One possible application scenario for our system is to integrate the visual similarity detection scheme into the open source tool AntiPhish [4, 7]. AntiPhish tracks the sensitive information of a user and generates warnings whenever the user attempts to provide this information on a web site that is considered to be untrusted. It works in a fashion similar to a form-filler application. However, it not only remembers what information (i.e., a username, password pair) a user enters on a page, but it also stores where this information is sent to. Whenever a tracked piece of information is sent to a site that is not in the list of permitted web sites, AntiPhish intercepts the operation and raises an alert. Although simple, the approach is effective in preventing phishing attacks. Unfortunately, when a user decides to reuse the same username, password pair for accessing different online services, too many undesired warnings (i.e., false positives) are raised. By integrating the comparison technique into the existing AntiPhish solution, we can prevent AntiPhish from raising warnings for sites that are visually different. The underlying assumption is that a phishing page aims to mimic the appearance of the targeted, legitimate page. Thus, when two pages are similar, and the user is about to enter information associated with the first page on the suspicious, second page, an alert should be raised. When the two pages are different, it is unlikely that the second page tried to spoof the legitimate site, and thus, the information can be transmitted without a warning. Of course, our technique can also be used in other application scenarios, as long as a baseline for the suspicious page is available. That is, we need to know what the legitimate page looks like so that we can compare against it. For example, the approach could be part of a security solution that works at the mail server level. Whenever a suspected phishing email is found, the potential phishing URL is extracted from the email. Then, the corresponding legitimate page is obtained, using a search engine or, based on keywords, selected among a predefined set of registered pages. Finally, a comparison is initiated and, if the outcome is positive, the email is blocked. To compare a target page (i.e., suspected page) with a legitimate page, four steps are required:

1. Retrieve the suspicious web page w.
2. Transform the web page into a signature S(w).
3. Compare S(w) with the stored signature S(ˆ w) of the supposed legitimate page ˆ w (i.e., the page targeted by the phishing page).
4. If the signatures are “too” similar, raise an alert.

Steps 2 and 3 represent the core of our technique. We discuss these steps in detail in the next two subsections. The actual implementation of Step 4 depends on the specific application scenario in which the approach is used. For example, in Antiphish, raising an alert implies that the submission of sensitive data is canceled and a warning is displayed to the user.

3.2 SIGNATURE EXTRACTION

A signature S(w) of a web page w is a quantitative way of capturing the information about the text and images that compose this web page. More precisely, it is a set of features that describe various aspects of a page. These features cover

(i) each visible text section with its visual attributes,
(ii) each visible image, and
(iii) the overall visual look-and-feel (i.e., the larger composed image) of the web page visible in the viewport1, as follows.

Concerning visible text, we consider each visual piece of text on the web page that corresponds to a leaf text node in the HTML DOM tree and we extract, for each one: its textual content, its foreground color, its background color, its font size, the name of the corresponding font family, and its position in the page (measured in pixel starting from the upper left corner). For each visible image of the web page, our technique extracts: the value of the corresponding src attribute (i.e., the source address of the image), its area as the product of width and height, in pixel, its color histograms, its 2D Haar wavelet transformation, and its position in the page. The 2D Haar wavelet transformation [11] is an efficient and popular image analysis technique that, essentially, provides low-resolution information about the original image.

Finally, we consider the overall image corresponding to the viewport of the web page, as rendered by the user agent, and we extract its color histograms and its 2D Haar wavelet transformation.

3.3 SIGNATURE COMPARISON

Once two signatures S(w) and S(ˆ w) are available, we can compute the similarity score between the corresponding web pages w and ˆ w. To this end, we start by comparing pairs of elements from each page. Of course, elements are only compared with matching types (e.g., text elements are only compared with other text elements). That is, we compare all pairs of text elements to obtain a similarity score s^t. Then, we compare all image pairs to obtain a similarity score s^i.Finally, the overall appearances of the two pages are used to derive a similarity score so. Using these three scores, a single similarity score $s \in [0, 1]$ is derived that captures the similarity between the pages w and ˆ w. Due to space constraints, we omit a detailed discussion on the comparison of elements. For more details, the reader is referred to [5]. In summary, a pair similarity index is output by each pair comparison, which takes into account the features of the pair elements specified in the previous section. The text elements comparison involves computing the Levenshtein distance between the two corresponding strings, the 1-norm distance between the foreground and background colors, and the Euclidean distance between the positions in the page. Furthermore, we consider font families and sizes too. The image elements comparison involves computing the Levenshtein distance between the two corresponding src attributes, the 1-norm distance between the color histograms and the 2D Haar wavelet transformations, and the Euclidean distance between the positions in the page. We also consider image areas. The same strategy, except for the image areas and page positions, is followed for the overall image.

The viewport is the part of the web page that is visible in the browser window. Once we have obtained a pair similarity index for each pair of elements, we store them in a similarity matrix: St for text elements and Si for images the overall image comparison outputs a single similarity score so, hence no matrix is needed. For ease of reasoning, we discuss only the text elements matrix St: the same information applies to Si. The dimension of the matrix is n×m, where n is the number of text elements on page w and m is the number of elements on ˆ w.

To obtain a similarity score s from a similarity matrix S (s^t for the text matrix St and s^i for the image matrix Si), we average the largest n elements of the similarity matrix, which are selected using the following iterative, greedy algorithm:

(i) we select the largest element of the matrix;

(ii) we discard the column and the row of the selected element. We repeat these steps until a number n of elements are selected or the remaining matrix is composed of either no rows or no columns. We set n = 10 for the text similarity matrix and n = 5 for the image similarity matrix. In other words, we extract the n most matching items (either among text blocks or among images) between the two web pages under comparison, avoid to consider an item more than once.

We consider the average of the greatest n values in the matrix instead of considering the whole matrix because we wish to avoid the case in which the comparison outcome is influenced mainly by many non-similar elements rather than by few, very similar elements (which are typically the ones that can visually lure the user) For example, consider a phishing page in which there are very few images (e.g., the logo and a couple of buttons) that are very similar to the ones in the legitimate page. Also, imagine that there are a large number of graphical elements, possibly small and actually rendered outside of the viewport, which are not present in the original page; if we would take the average over all the matrix elements, the outcome would be biased by the low similarity among the many dissimilar elements. However, the user would be tricked by the few elements that are very similar. The final outcome s of a comparison between two signatures is obtained as $s = a^t s^t + a^i s^i + a^o s^o$. When s is large, the two pages are similar. A threshold t is used in order to discriminate between the two cases: w and ˆ w are considered similar if and only if $s \geq t$, not similar otherwise. We discuss how we determine suitable values for the coefficients a^t, a^i, a^o as well as for the threshold t in Section 4.2.

4. EXPERIMENTAL EVALUATION

In this section, we discuss the experiments that we performed to demonstrate the effectiveness of our system to recognize similarities between phishing pages and their targets (i.e., the legitimate pages that are spoofed).

4.1 DATASET

First, we compiled a dataset that consists of negative and positive pairs of web pages. For the positive pairs, we selected pairs of real-world legitimate pages and corresponding phishing pages. We obtained the phishing pages from the PhishTank public archive (http://www.phishtank.com). For each phishing page, we retrieved the corresponding legitimate page by visiting the web site of the spoofed organization immediately after the attack appeared on PhishTank. To build the negative part of the dataset, we collected a number of common web pages, unrelated to the legitimate
ones.

LEVELS	Fp	Nn	FPR	Fn	Np	FNR
All (0,1 and 2)	0	140	0%	2	27	7.4%
Only 0 and 1	0	140	0%	0	20	0.0%
Only 0	0	140	0%	1	1	0.0%

Table: FPR and FNR for different datasets
FPR-false positive rate
FNR-false negative rate

We partitioned the set of positive pairs into three subsets, based on their visual similarity, as perceived by a human viewer: Level 0 identifies pairs with a perfect or almost perfect visual match. Level 1 identifies pairs with some different element or with some minor difference in the layout. Level 2 identifies pairs with noticeable differences. We chose to partition positive pairs into different subsets for the following reason. The majority of phishing pages exactly mimic the appearance of the legitimate page. This is not surprising, as the miscreants do not wish to raise suspicion.

However, there are also cases where visual differences do exist. These differences may be simply due to the poor skills of the attacker (e.g., mistakes in a text translated to a foreign language). However, some differences may be voluntarily inserted, either at the source level or at the rendering level. This could be done to evade anti-phishing systems, while, at the same time, keeping the look-and-feel as close to the original web page as possible. Note that similar

evasion techniques are sometimes used by spammers for image-based spam [2]. That is, although some randomized alterations are applied to the original image, from the user's point of view, the image remains identical.
We also partitioned the set of negative pairs into two subsets. One subset consists of banking web pages with a login form; elements in the second subset have no such forms and vary in size, layout, and content. We chose to include a substantial portion of pages with a login form to make the experiments more realistic and challenging. The dataset was composed of 41 positive pairs (20 of Level 0, 14 of Level 1, and 7 of Level 2). We had 161 negative pairs (115 with and 46 without a login form).

4.2 TESTING METHODOLOGY

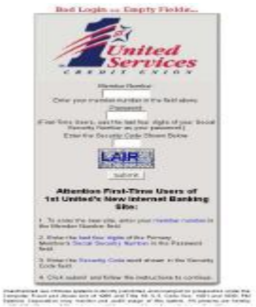
Using our dataset, we built a training set by extracting a small portion of pairs of pages. This subset was used to tune the coefficients a^t, a^i, a^o and the threshold t used in the computation of the final similarity score: we found the optimal values by minimizing a function of FPR and FNR on the training set using an implementation of the simplex method. The training set was composed of 14 positive pairs (9 of Level 0, and 5 of Level 1) and 21 negative pairs (15 with and 6 without a login form).

4.3 RESULTS

We then evaluated our approach using the parameter values a^t, a^i, a^o and the threshold t computed as explained above over the remaining part of the dataset. For this experiment, the test set was composed of 27 positive pairs (11 of Level 0, 9 of Level 1, and 7 of Level 3) and 140 negative pairs (100 with and 40 without a login form).

Table 1 summarizes the results of the tests. It can be seen that our approach detects all phishing pages classified as Level 0 and 1, while it fails to detect two out of seven positive pairs of Level 2. Hence, we exhibit an overall false negative rate (FNR) equal to 7.4%. We verified, by visual inspection, that those two positive pairs were indeed difficult to detect by our visual-similarity-based approach.

Figure 1 shows screenshots of one of the two undetected pairs; note that both the overall appearance and textual contents of the pages are significantly different.



Bad Login -- Empty Fields...

Member Number

Enter your member number in the field above.

Password

(First-Time Users, use the last four digits of your Social Security Number as your password.)

Enter the Security Code Shown Below

Attention First-Time Users of 1st United's New Internet Banking Site:

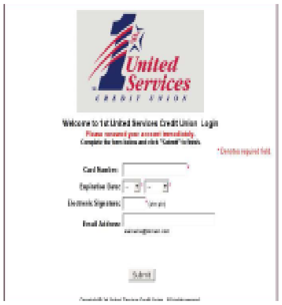
1. To enter the new site, enter your member number in the Member Number field.

2. Enter the last four digits of the Primary Member's Social Security Number in the Password field.

3. Enter the Security Code word shown in the Security Code field.

4. Click submit and follow the instructions to continue.

[illegible]



Welcome to 1st United Services Credit Union Login

Please reconcel your account immediately.

Complete the form below and click "Submit" to finish.

* Denotes required field.

Card Number: *

Expiration Date: *

Electronic Signature: *

Email Address:

fig 1: legitimate web page Fig 2:Phishing attempt level 2

Figure 1and figure 2 : One of the two missed positive pairs (Level 2). Note that the phishing page is visually significantly different from the legitimate one. Also, Table 1 results show that our approach does not raise any false positive on the 140 negative pairs, which results in a false positive rate (FPR) of 0%. This includes all the negative pairs corresponding to pages containing a login form. We also computed FPR and FNR for the same three dataset compositions for various values of the threshold t.

In Figure 3 FPR and FNR are plotted as functions of the threshold t; FNR is shown for subsets of the test set including and not including, respectively, positive pairs of Level 2. One can see that there exists at least one threshold value for which our approach exhibits perfect behavior when considering a dataset which does not contain Level 2 positive pairs. Moreover, our approach is robust to certain variations of t.

In general, the choice of t depends on the desired trade-off between possible false positives and possible false negatives. Hence, it depends on the context in which the proposed approach is supposed to be deployed. For example, if we implement the comparison approach as part of the AntiPhish tool, it may be preferable to select lower values for t. The reason is that when using the visual comparison component with Anti Phish, a large number of possible false positives is already filtered out, since the comparison is invoked only when a user's known credentials are about to be transmitted to an untrusted web site. Therefore, the comparison may be relaxed towards accepting more false positives (warnings) in favor of avoiding missed detections.

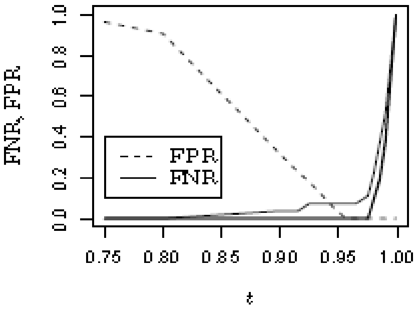


Fig 3: false positive/negative rates vs t

Finally, we measured the average computation time for comparing two pages. Note that such an operation involves both the signature extraction and the signature comparison phase.

In our experimental analysis, we focused on the comparison phase. The extraction phase consists mainly of retrieving information which, in practice, is already available to the browser that has rendered the page. Moreover, the legitimate page signature is typically extracted once at a previous point in time. In our experiments, we found that it took about 3.8 seconds for positive pairs and about 11.2 seconds for negative pairs to be compared. These numbers were obtained on a dual AMD Opteron 64 with 8GB RAM running a Linux OS. Note that the comparison of two signature $S(w)$ and $S(\hat{w})$, as described in Section 3.3, requires a number of operations in the order of $m \cdot \hat{m}$, where m and $\hat{m}$ are the corresponding number of tuples. Clearly, waiting for more than 10 seconds for a single comparison is prohibitive.

To address this problem, we implemented the following optimizations that result in a considerable reduction of computational costs. The key idea to improve performance is to execute the comparison operations in an order such that the most expensive operations are executed last. In particular, during the similarity index computation (either st, for the text section, or si, for the images), we keep the n (with n = 10 for the text part and n = 5 for the image part) similarity values for the most similar pairs of tuples found so far. When evaluating a new pair of tuples, we can stop the evaluation once we determine that this pair cannot exceed the similarity values of one of the top n pairs, even if all remaining features comparisons yield perfect similarity. For example, suppose that: (i) we are considering a pair of images, (ii) we have computed the distance in terms of positions on the page and image sizes, and (iii) the distances are such that the corresponding matrix element will be lower than the 5th greatest element of the matrix. In this case, we do not need to compute, nor extract, the two Haar transformations or the Levenshtein distances.

A similar optimization is performed based on the outcome of the overall image comparison. Once we determine that, after looking at the score for the overall appearance, two pages cannot exceed the similarity threshold t, and then we do not need to compute any of the two similarity matrices for the text and image elements. This results in impressive speed-ups. A negative comparison between two pages is produced in a few milliseconds.

5. CONCLUSION

In this paper, we presented an effective and novel approach to detect phishing attempts by comparing the visual similarity between a suspicious page and the potential, legitimate target page. The proposed approach is inspired by two previous open source anti-phishing solutions: the AntiPhish browser plugin and its DOMAntiPhish extension. Our solution addresses the shortcomings of these approaches and aims to make these systems more effective. When checking for visual similarity, we consider three page features: text pieces, images embedded in the page, and the overall visual appearance of the web page as rendered by the browser. We consider features that are visually perceived by users because, as reported in literature, victims are typically convinced that they are visiting a legitimate page by judging the look-and-feel of a web site.

We performed an experimental evaluation of our comparison technique to assess its effectiveness in detecting phishing pages. We used a dataset containing 41 real phishing pages with their corresponding target legitimate pages. The results, in terms of false alarms and missed detection, are satisfactory. No false positives were raised and only two phishing attempts (that actually did not resemble the legitimate web page) were not detected.

7. REFERENCES

[1] APWG. Phishing Activity Trends - Report for the Month of December, 2007. Technical report, Anti Phishing Working Group, Jan. 2008.

[2] H. Aradhye, G. Myers, and J. Herson. Image analysis for efficient categorization of image-based spam e-mail. Document Analysis and Recognition, 2005.

[3] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. Mitchell. Client-side defense against web-based identity theft..

[4] E. Kirda and C. Kruegel. Protecting Users against Phishing Attacks. The Computer Journal, 2006.

[5] E. Medvet, E. Kirda, and C. Kruegel. Visual-Similarity-Based Phishing Detection. Technical Report,

[6] Microsoft. Sender ID Home Page. http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx, 2008.

[7] A. Rosiello, E. Kirda, C. Kruegel, and F. Ferrandi. A Layout-Similarity-Based Approach for Detecting Phishing Pages. IN (SecureComm), 2007.

[8] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. C. Mitchell. A Browser Plug-In Solution to the Unique Password Problem.

[9] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. C. Mitchell. Stronger Password Authentication Using Browser Extensions. In 14th Usenix Security Symposium, 2005.

[10] SpoofGuard. Client-side defense against web-based identity theft.

[11] R. Stankovic and B. Falkowski. The Haar wavelet transform: its status and achievements. Computers and Electrical Engineering, 29:25–44, 2003.