

Defending the weakest link: phishing websites detection by analysing user behaviours

Xun Dong · John A. Clark · Jeremy L. Jacob

Published online: 18 February 2010
© Springer Science+Business Media, LLC 2010

Abstract Phishing detection systems are principally based on the analysis of data moving from phishers to victims. In this paper we describe a novel approach for detecting phishing websites based on analysis of users' online behaviours—i.e., the websites users have visited, and the data users have submitted to those websites. Such user behaviours can not be manipulated freely by attackers; detection based on those data can achieve high accuracy whilst being fundamentally resilient against changing deception methods.

Keywords Phishing attacks · Phishing websites detection · Identity theft · User protection

1 Introduction

Phishing attacks are well-organised and financially motivated crimes which steal users' confidential information and authentication credentials. They not only cause significant financial damage to both individuals and companies/financial organisations, but also damage users' confidence in e-commerce as a whole. According to Gartner analysts, financial losses stemming from phishing attacks have risen to more than 3.2 billion USD with 3.6 million victims in 2007 in the US [19], and consumer anxiety about Internet security resulted in a two billion USD loss in e-commerce and banking transactions in 2006 [18].

The scale and sophistication of phishing attacks have been increasing steadily despite numerous countermeasure efforts. The number of reported phishing web sites increased five-fold from 10047 to 55643 in the 10 month period between June 2006 and April 2007.¹ During the 2008 world financial crisis, phishing attack incidents have increased three times compared to the same period in 2007. The real figure may be much higher because many sophisticated phishing attacks (such as context aware phishing attacks, malware based phishing attacks, and real-time man-in-the-middle phishing attacks against one-time passwords [30]) may not all have been captured and reported.

In this paper we present our design, implementation and evaluation of the *user-behaviour* based phishing detection system (UBPD). UBPD does not aim to replace existing anti-phishing solutions, rather it complements them. It alerts users when they are about to submit credential information to phishing websites (i.e. when other existing countermeasures have failed), and protects users as the last line of defence. Its detection algorithm is independent of how phishing attacks are implemented, and it can easily detect sophisticated phishing websites that other techniques find hard to deal with. In contrast to existing detection techniques based only on the *incoming* data, this technique is much simpler, needs to deal with much less low level technical detail, and is more difficult to bypass by varying spoofing techniques and deception methods.

Note: In the rest of the paper we use 'interact', 'interaction' and 'user-webpage interaction' to refer to the user supplying data to a webpage.

X. Dong (✉) · J.A. Clark · J.L. Jacob
Department of Computer Science, University of York, York, UK
e-mail: xundong@cs.york.ac.uk

J.A. Clark
e-mail: jac@cs.york.ac.uk

J.L. Jacob
e-mail: jeremy@cs.york.ac.uk

¹ Anti-phishing work group home page, 2007. <http://www.antiphishing.org/>.

2 Related studies

Researchers have tried to understand phishing attacks by studying the human factors in phishing, usability of the security features of the systems and the techniques used in phishing attacks. To fight against phishing attacks system designers have invented novel security interfaces, automated detection systems and education methods. In this section we briefly survey the major findings in this area.

2.1 What is phishing?

Jacobsson has introduced tools and models to describe a variety of phishing attacks in a uniform and compact manner. He has also presented an overview of potential system vulnerabilities and corresponding defence mechanisms [14].

Abad has presented a comprehensive process flow of a phishing attack [1]. The model includes almost every step of a phishing attack from attack preparation (creation or renting botnets, designing the phishing attacks) to turning the confidential information harvested into a financial profit.

Ollmann surveys the technologies and technical security flaws phishers exploit to conduct their attacks, and provides detailed advice on what organisations can do to prevent future attacks [25, 26]. Watson et al. review the actual techniques and tools used by phishers, providing three examples of empirical research where real-world phishing attacks were captured using honeynets [34].

2.2 Human factors in phishing attacks

Dhamija et al. have investigated why users fall victim to phishing attacks by carrying out a controlled phishing attack user study [6]. They identified three major causes: (1) a lack of understanding of how computer systems work; (2) a lack of attention to security; and (3) the high quality visual deception practised by the phishers.

Schechter et al. evaluated website authentication measures that are designed to protect users from man-in-the-middle, phishing, and other site forgery attacks and also investigated the impact of role playing on the accuracy of the study result [29]. Their findings are: (1) users will enter their passwords even when HTTPS indicators are absent; (2) users will enter their passwords even if site authentication images are absent; (3) site-authentication images may cause users to disregard other important security indicators; and (4) behaviour in role playing may not be representative of normal behaviours, because in such studies no real losses would be incurred.

Jakobsson et al. have studied what makes phishing emails and web pages appear authentic [16]. Elsewhere Jakobsson summarised comprehensively what typical computer users are able to detect when they are carefully watching for signs

of phishing [15]. The findings are: (a) spelling and design matter; (b) third party endorsements depend on brand recognition; (c) too much emphasis on security can backfire; (d) people look at URLs; (e) people judge relevance before authenticity; (f) personalisation creates trust; (g) emails are very phishy, web pages a bit, phone calls are not; (h) padlock icons have limited direct effects; and (i) independent channels create trust.

Jagatic et al. have shown how publicly available personal information from social networks (such as friendster, myspace, facebook, orkut, and linkedin) can be used to launch effective context aware phishing attacks [13]. In their studies, they first determine people's social networks and then masquerade as one of the victim's social contacts to create an email to the victim (using email header spoofing techniques). They examine how easily and how effectively a phisher can exploit social network data found on the Internet to increase the yield of a phishing attack. The answer, as it turns out, is: very easily and very effectively. Their study suggests that Internet users may be over four times as likely to become victims if they are solicited by someone appearing to be a known acquaintance.

2.3 Design guidelines

Wu et al. have discovered that the security tools such as security toolbars are not effective enough to protect people from falling victim to phishing attacks by conducting two studies of three security toolbars and other browser security indicators [35]. Based on their findings, the authors suggest that the alert should always appear at the right time with the right warning message; user intentions should be respected, and if users must make security-critical decisions they should be made consciously; and it is best to integrate security concerns into the critical path of their tasks so that users must address them.

2.4 Technology countermeasures

Many anti-phishing email filters have been invented to fight phishing at the email level, as it is the primary channel for phishers to reach victims. SpamAssassin,² PILFER [7], and Spamato [2] are typical examples of those systems. They apply predefined rules and characteristics often found in phishing emails to analyse incoming emails. PHONEY is a phishing email detection system that tries to detect phishing emails by mimicking user responses and providing fake information to suspicious web sites that request critical information. The web sites' responses are forwarded to the decision engine for further analysis [4].

²Spamassassin's official website. <http://spamassassin.apache.org/>.

Web Wallet [36] tries to create a unified interface for authentication. It scans pages for the login form. If such a form exists, then it asks the user to explicitly indicate his/her intended site to login. If the intention matches the current site, it automatically fills webpage input fields. Otherwise a warning will be presented to the user.

Internet Explorer [20], Firefox 2 with Google's safe browsing [23], and Netcraft toolbar [24] are all blacklist anti-phishing websites detection systems. These systems check whether the URL of the current web page matches any URL in a list of identified phishing web sites.

SpoofGurad[5] is a typical signature and rule based detection system. It detects phishing webpages by analysing the host name, URL, and the images used in the current webpage.

CANTINA [37] detects phishing websites based on the TF-IDF information retrieval algorithm. It retrieves the key words of the current webpage and uses Google search to analyse if the current webpage is in the top 30 or not.

Ying Pan et al. have invented a phishing website detection system which examines the anomalies in web pages, in particular, the discrepancy between a web site's identity and its structural features and HTTP transactions [27].

3 Design

3.1 Phishing nature and detection philosophy

Phishing websites work by impersonating legitimate websites, and they have a very short life time. On average a phishing website lasts 62 hours, and users rarely visit the phishing website prior to the attack [22].

Secondly, phishing attacks always generate mismatches between a user's perception and the truth. In successful web based phishing attacks, victims have believed they are interacting with websites which belong to legitimate and reputable organisations or individuals. Thus the crucial mismatch that phishers create is one of real *versus* apparent identity.

Phishing attacks can be detected if we can detect such a mismatch. One approach is to predict a user's perception and then compare it with the actual fact understood by the system. CANTINA is an example of this approach [37]. The main issue with this approach is that the data the system relies on is under the control of attackers, and there are so many techniques that attackers can apply to manipulate the data to easily evade detection. For CANTINA, attackers could use images instead of text in the body of the webpage, they could use iframes to hide a large amount of content from the users while computer programs can still see it; they could use Javascript to change the content of the page after the detection has been done.

We decide to use another, more reliable, approach. The authentication credentials, which phishers try to elicit, ought to be shared only between users and legitimate organisations. Such (authentication credential, legitimate website) pairs are viewed as the user's *binding relationships*. In legitimate web authentication interactions, the authentication credentials are sent to the website they have been bound to. In a phishing attack the mismatches cause the user to unintentionally break binding relationships by sending credentials to a phishing website. No matter what spoofing techniques or deception methods are used, nor how phishing webpages are implemented, the mismatch and violation of the binding relationships always exists. So one can discover the mismatch by detecting violation of users' binding relationships.

Hence phishing websites can be detected when both of the following two conditions are met: (1) the current website has rarely or never been visited before by the user; (2) the data, which the user is about to submit, is bound to website other than the current one.

3.2 System design

Below we describe the design of the UBPD, providing an overview of how information flows through the system, how user behaviour related information is created and maintained, how the information is used to determine the risks of the current interaction, how phishing sites are detected, and how the security of the system is maintained.

3.2.1 Overview of the detection work flow

UBPD has three components:

- The user profile contains data to describe the user's binding relationships and the user's personal whitelist. The profile must be constructed before the system can detect phishing websites.
- The monitor collects the data the user intends to submit and the identity of the destination websites, and activates the detection engine.
- The detection engine uses the data provided by the monitor to detect phishing websites and update the user profile if necessary.

UBPD has two working modes: training mode and detection mode. In training mode, UBPD runs quietly in the background, and focuses on learning newly created binding relationships or updating the existing binding relationships. When in detection mode, UBPD checks whether any of the user's binding relationships would be violated if the user-submitted data is sent to the current website. The mode in which UBPD runs is decided by checking whether the webpage belongs to a website

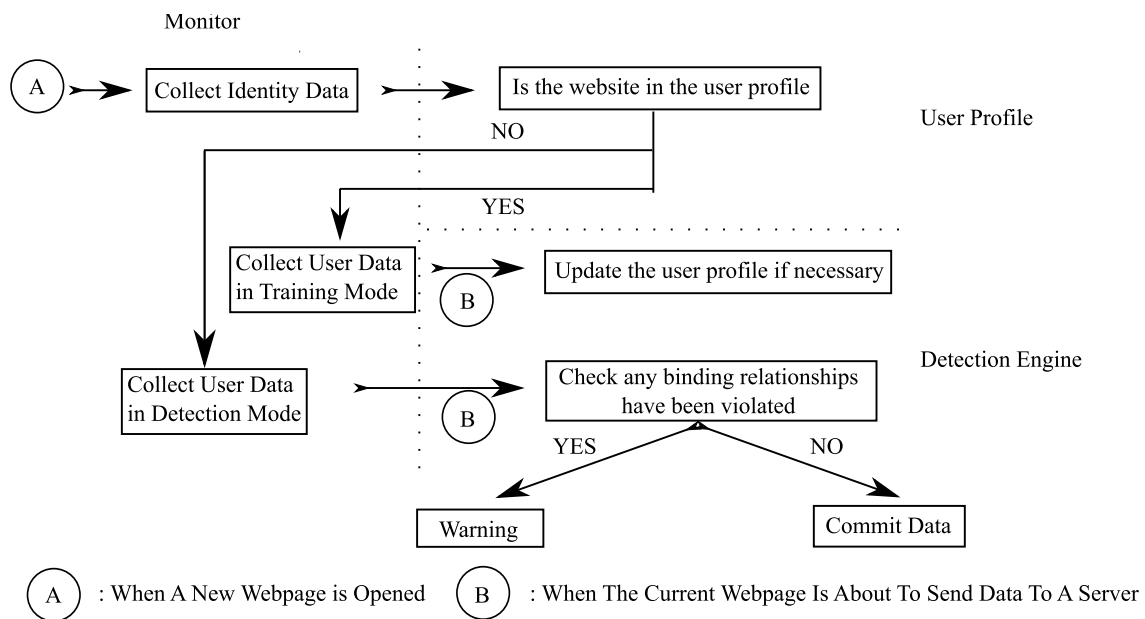


Fig. 1 Detection process work flow

1. whose top level domain³ is in the user's personal white list, or
2. with which the user has shared authentication credentials.

If either is true the system will operate in training mode, otherwise, in detection mode. Phishing webpages will always cause UBPD to run in the detection mode, since they satisfy neither condition.

The detection work flow is shown in Fig. 1. Once a user opens a new webpage, the monitor decides in which mode UBPD should run. Then, according to the working mode the monitor chooses an appropriate method to collect the data the user submitted to the current webpage, and sends it to the detection engine once the user initiates data submission. The details of the data collection methods are discussed in Sect. 4. When running in detection mode if the binding relationships are found to be violated, the data the user submitted will not be sent and a warning dialogue will be presented. For the remaining cases, UBPD will allow data submission.

3.2.2 Creation of the user profile

The user profile contains the user's binding relationships and personal whitelist. The binding relationships are represented as a collection of paired records, i.e., $\langle aTopLevelDomain, aSecretDataItem \rangle$. The personal whitelist is a list of top level domains of websites.

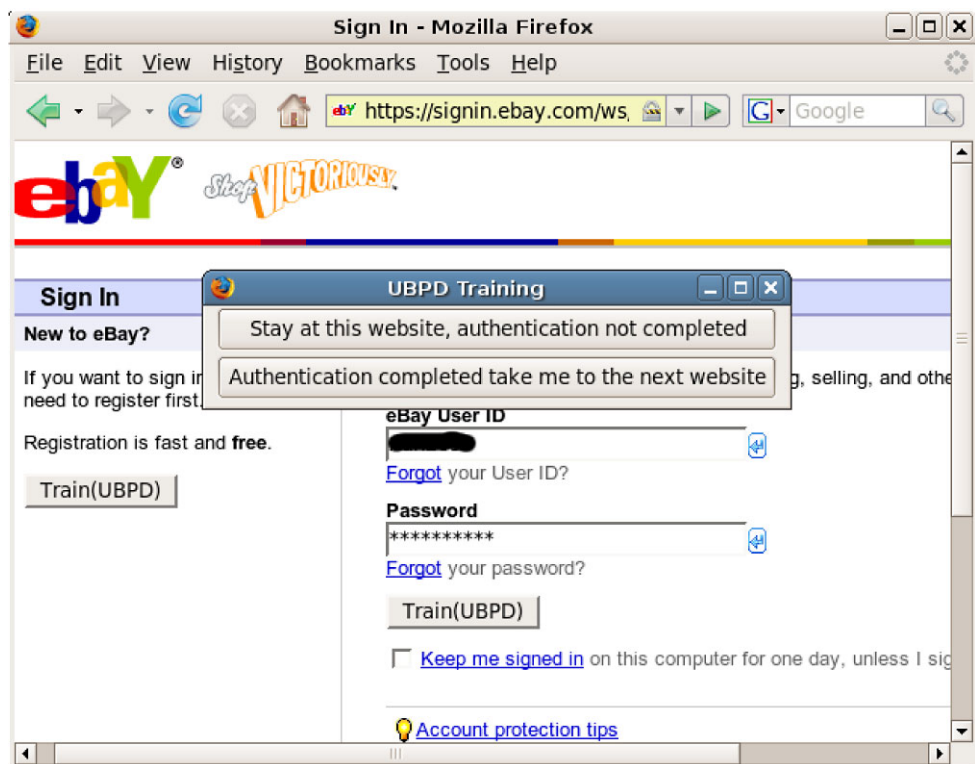
Creation of the binding relationship records is a mandatory step of UBPD's installation procedure. Having installed

the UBPD, when the browser is restarted the user is presented with a dialogue which asks for the websites with which the user has accounts. Then UBPD automatically takes users to those websites, and asks users to fill in the authentication forms on those websites one by one. In addition, UBPD also dynamically modifies each authentication webpage, so that all the buttons on the webpage will be replaced with the 'Train(UBPD)' button. Once a user clicks on it, UBPD creates new binding relationship records in the user profile and then asks whether the user wants to carry on the authentication process (in case it is a multiple step authentication) or go to the next website. A screen shot is shown in Fig. 2.

On average users have over 20 web accounts [8]. We do not expect them to train UBPD with all their binding relationships, however, we do expect them to train UBPD with their most valuable binding relationships (such as their on-line accounts with financial organisations). Our preliminary evaluations have confirmed that our assumption is valid.

There are two types of binding relationships that UBPD is unaware of: the ones that already exist but users have not trained UBPD with, and the ones that users will create in the future. It is possible that the authentication credentials in these unknown binding relationships have also been used in the binding relationships that UBPD is aware of (as the result of authentication credentials reuse). If UBPD were to make a decision based only on available binding relationships information, then false alarms would be generated. To avoid such false alarms, the whitelist is automatically created to include the websites that users have already got accounts with but did not make UBPD aware of and the web-

³Suppose the URL of a webpage is "domain2.domain1.com/files/page1.htm", the top level domain is "domain1.com".

Fig. 2 User profile creation

sites they may have accounts with in future. As stated in Sect. 3.2 if the website with which the user is currently interacting is found to be in the whitelist UBPD will only run in training mode. In the training mode, such reuse of the authentication credentials can only cause UBPD to either create new binding relationships record or update existing ones.

The personal whitelist is constructed by combining a default whitelist with the websites the user has visited more than three times (configurable) according to the user's browsing history. The default whitelist is constructed by identifying the 1000 websites that are most visited in the user's country. This information is obtained from Alexa, a company specialising in providing Internet traffic ranking information. The default whitelist could be viewed as a reflection of mass users' online browsing behaviours. These most popular websites are not phishing websites, and users are likely to have binding relationships with them in the future if they do not have them already. In this way, websites that users currently have accounts with or are going to have account with are very likely to be included in the whitelist. By default, this whitelist is automatically updated weekly. It is worthy noting that, the whitelist in UBPD is mainly used for reducing false warnings, it is not used as a conventional whitelist to decide whether the current website is legitimate or not.

3.2.3 Update of the user profile

In addition to the manual update by users, when running in the training mode UBPD has an automatic method to update the user profile with the other currently unknown binding relationships. It detects whether the user is using a web authentication form. This is achieved by analysing the HTML source code, such as the annotation, label, use of certain tags (such as <form>) and type of the HTML elements. If the user is using a web authentication form, and the user profile contains no binding relationships with the current website, then UBPD prompts a window to ask the user to update the user profile. If there is an existing binding relationship for the current website, then UBPD will replace the authentication credentials in the binding relationships with the latest values the user submits. If users have entered the authentication credentials wrongly, those credentials will still be stored, but those wrong values will be corrected, when users relog in with the correct authentication credentials. In future the detection of web authentication page usage can be much simpler and more accurate once web authentication interfaces [11, 31, 33] are standardised. Current authentication form detection is still accurate because it deals with legitimate websites (automated update is carried out only in training mode), and those legitimate websites use standard web forms and will not deliberately disrupt detection.

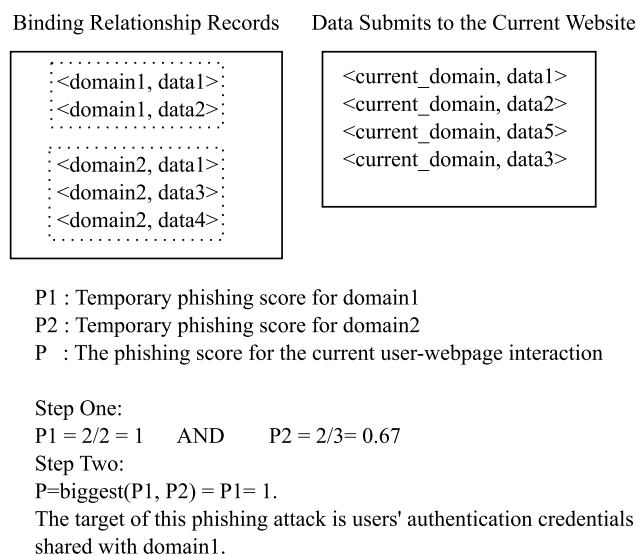


Fig. 3 An example of how the phishing score is calculated

3.2.4 Phishing score calculation

In detection mode, UBPD decides whether the current webpage is a phishing webpage by calculating phishing scores. The calculation is a two step process. In the first step, for each legitimate website, with which the user has shared authentication credentials, a temporary phishing score is calculated. Each temporary phishing score is the fraction of the authentication credentials associated with a legitimate website that also appear in the data to be submitted to the current webpage. Its value ranges from 0.0 to 1.0.

In the second step, those temporary phishing scores are sorted into descending order. The current webpage's phishing score is the highest score calculated. The legitimate website with the highest temporary phishing score is considered to be the impersonated target of the phishing website. If more than one legitimate website has yielded the highest phishing score (due to the reuse of authentication credentials), they will all be considered as targets. Although it may not be the attacker's intent, the data they get if an attack succeeds can certainly be used to compromise the user's accounts with those legitimate websites. Figure 3 illustrates how a phishing score is calculated in UBPD.

Given our phishing score calculation method, clever attackers may ask the user victims to submit their credential information through a series of webpages, with each phishing webpage asking only for a small part of data stored in the user profile. To handle this fragmentation attack UBPD has a threshold value and cache mechanism. The system maintains a history of which shared credentials have been released and so when a credential is about to be released an accumulated score can be calculated. Once the phishing score is above the threshold the current webpage will be considered as a phishing webpage. The system's default

threshold is 0.6. Why we choose 0.6 is discussed in Sect. 5. If the current phishing score is not zero, UBPD also remembers which data has been sent to the current website and will consider it in the next interaction if there is one. Accordingly many fragmentation attacks can be detected.

3.2.5 Reuse

It is very common that a user shares the same authentication credentials (user names, passwords, etc.) with more than one website. The two running modes and the user's personal whitelist are designed to prevent false warnings caused by reuse without compromising detection accuracy.

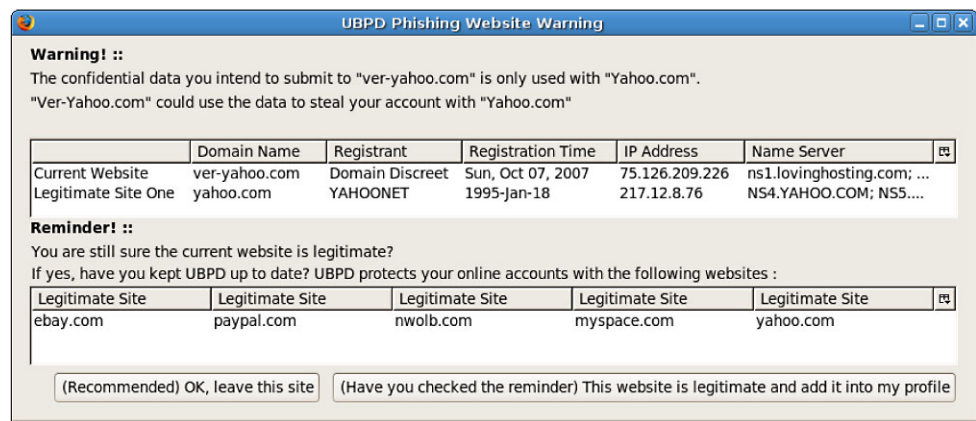
UBPD detects the violation of binding relationships only when the user is interacting with websites that are neither in the user's whitelists nor which the user has account with. So long as legitimate websites for which users have used the same authentication credentials are all contained in the user profile, there will be no false phishing warning generated due to the reuse. The method that UBPD uses to create the user profile ensures such legitimate websites are most likely to be included, as those websites are either within the user's browsing history or are popular websites in the user's region. Our preliminary false positive evaluation also confirmed this.

3.2.6 Warning dialogue

The content of the warning dialog must be suitable for users with very limited knowledge, otherwise, as a previous study [35] found, users may ignore the warning or may not behave as suggested. Figure 4 is a warning dialog example. To make the information easy to understand, the dialogue tells users that the current website, to which they are submitting credentials, is not one of the legitimate websites associated with those authentication credentials. To help users understand the detection result and make a correct decision, UBPD also provides information regarding the differences between the legitimate website and the possible phishing website in five aspects: the domain name, the domain registrant, the domain registration time, name servers, and IP addresses. Users don't need to understand those terms. They only need be able to recognise the difference between values of the five attributes of the legitimate website and the phishing website.

3.2.7 Website equivalence

To discover whether the user is about to submit the authentication credentials to entities with which they have not been bound, UBPD needs to be able to accurately decide whether a given website is equivalent to the website recorded in user's binding relationships. It is much more complicated

Fig. 4 Phishing warning dialogue

than just literally comparing two URLs or IP addresses of two websites, because: (1) big organisations often have web sites under different domain names, and users can access their account from any of these domains; (2) the IP address of a website can be different each time if dynamic IP addressing is used; (3) it is hard to avoid ‘pharming’ attacks, in which the phishing site’s URL is identical to legitimate one.

Our system first compares two websites’ domain names and IP addresses. When the two domain names and two IP addresses are equal the web sites are assumed to be identical. Otherwise, the system interrogates the WHOIS⁴ database and uses the information returned to determine equivalence. When analysing two different IP addresses our system compares the netnames, name servers, and the countries where each IP address is registered. If they are all identical then the two websites are deemed to be identical. This method can also detect pharming attacks, in which both fraudulent and legitimate websites have the same domain name but are hosted on different IP addresses.

This method is not perfect. A more elegant and complete solution would be a mechanism where servers provide security relevant metadata (including the outsourcing information) to the web browser via a standard protocol as suggested by Behera and Agarwal [3]. However, unless it becomes a standard we have to rely on WHOIS. The extended validation certificate [9] can provide information to decide whether two websites belong to the same party, but the problem is the high cost, high entry level and complexity of obtaining such certificates. Many small and medium businesses will not be able to own them; as a result they cannot be used to decide website equivalence in general.

⁴WHOIS is a TCP-based query/response protocol which is widely used for querying a database in order to determine the owner of a domain name, an IP address. RFC 3912 describes the protocol in detail.

3.2.8 User’s privacy

Since a user profile contains confidential user information, it is important that it does not add new security risks. We use a one-way secure hash function to hash the confidential data before it is stored. When the system needs to determine the equivalence between the data, the system just needs to compare the hash values.

However, because the domain name of the website is not hashed, if the profile is obtained by attackers they would be able to find out with which websites users have accounts and where users have reused their authentication credentials. This information is helpful for attackers; for example, it can be used to launch context aware phishing attacks against the users. To prevent this, when the system is installed it will randomly generate a secret key, and this key will be used to encrypt and decrypt the domain names.

3.3 Implementation

UBPD is implemented as a Firefox add-on. Most of the detection is implemented using Javascript. The hash function we use is SHA-1 [17] and the encryption method we use is Twofish.⁵ The hash function and encryption methods can be changed, we choose to use them mainly because there are open source implementations available. The user interface of the system is implemented by using XUL, a technology developed by Mozilla [10]. Although only implemented on Firefox, the detection system can be easily developed to work on other browsers, such as Internet Explorer and Opera.

⁵<http://home.versatel.nl/MAvanEverdingen/Code/>.

4 Evasion and countermeasures

4.1 Disruption of data collection

To accurately detect phishing UBPD needs to be able to work with the exact data the user is submitting to the current webpage. This data can be easily retrieved by accessing the DOM interface, however, the client script language can also manipulate the data before the monitor retrieves it.

In training mode, websites that users are visiting are legitimate, the monitor uses the DOM interface to retrieve the data, as in those cases client scripts manipulation is not an issue. However, in detection mode, when the user is interacting with websites that may be phishing websites, client scripts manipulation is a real threat. The monitor takes hence a different approach. The monitor performs like a keylogger, it collects user data by listening to a user's keystrokes for each element in the webpage. Such key logging is very efficient to run and since UBPD runs in detection mode only occasionally, there is little performance overhead.

4.2 Activation of the detection engine and denial of service attack

The detection engine must be activated before the data the user entered has been sent to a remote server. Normally webpages forward the user submitted data using built-in functions, such as the one provided by the standard web form. (Almost every legitimate website uses this function to submit the data.) So the detection engine is triggered and data submission is suspended when the monitor discovers the use of such functions. This is achieved by listening for 'DOMActivate' [12] events. These events are fired by the browser when such built-in functions have been used. This is the mechanism the monitor uses to activate the detection engine when the system is running in training mode.

However, phishers can use client side scripts to implement these built in functions. For example, by using Javascript they can access the DOM interface to retrieve the data users entered and use AJAX (Asynchronous Javascript And XML) techniques to send the data to the server before the user clicks the submit button. To prevent this evasion, UBPD could monitor the client script function calls when it is running under the detection mode. The function calls should be monitored are 'open()' and 'send()' from the 'xmlhttprequest' API [32]. These are the only two functions that client scripts can use to send data to the server. Once the monitor discovers such function calls, the function is suspended and the detection engine is triggered. Regardless of the data the client scripts would like to send, the detection engine always works on all the data the user has entered to the current webpage. The function call is only resumed if the detection engine thinks the current website is legitimate.

Since the detection engine would be triggered once the 'xmlhttprequest' API has been called, the attackers can issue this function call continuously to freeze the browser. To prevent this, the monitor can keep a record of whether there is user input since the last time the detection engine was activated for the same webpage. If there is then the detection engine will be activated, otherwise, not.

We plan to implement such client side scripts function call detection by using the techniques in Venkman (a Firefox JavaScript Debugger Addon) in the next version of UBPD. Nevertheless, if attacked by this evasion technique, at least UBPD would still be able to inform users what they have done, and preserve the evidence for possible analysis. One of the major problem for users is they do not know when they have been attacked and how. It gives users a good chance to reduce the damage caused by the phishing attacks.

4.3 Denial of service attack

Since the detection engine would be triggered once 'xmlhttprequest' has been called, the attackers can issue this function call continuously to freeze the browser. To prevent this the monitor will keep a record of whether there is user input since the last time the detection engine has been triggered for the same webpage. If there is then the detection engine will be activated, otherwise, not.

Our detection system decides whether two websites belong to the same party by analysing the information provided by the WHOIS database. In the current implementation we use only the WHOIS database, freely available on the Internet. The attackers could apply Denial of Service attacks on the WHOIS database we use so that the system would hang. This problem could be solved by using suitable network intrusion detection systems.

4.4 Insertion and fragmentation

Our detection system would work perfectly if the phishing webpage asks for the exact information stored in the user profile. However, attackers may try to vary the amount of information they request to disrupt the detection. Below are the two possible methods they might use.

Insertion. The information, that the phishing webpage asks for, includes not only the data contained in the user profile, but also includes data that does not.

Fragmentation. In this phishing attack, users will be asked to submit information through a series of webpages, and each phishing webpage asks for only a small part of data stored in the user profile.

The current phishing score calculation method (described in Sect. 3.2) automatically solves the insertion attacks. The

Table 1 Characteristics of user profile

	Alice	Bob	Carol	Dave
Reuse	No	No	Yes	Yes
Uniqueness	Strong	Weak	Strong	Weak

problem now is how to avoid the fragmentation attacks. The threshold value can be used to catch the fragmentation attacks. Once the phishing score is above the threshold the current webpage will be considered as a phishing webpage. The system's default threshold is 0.6. If the current phishing score is not zero, the system can remember which data has been sent to the current website (websites are recognised by their domains) and will consider it in the next interaction if there is one. Accordingly many fragmentation attacks can be detected as well.

5 Evaluation

We have carried out two experiments to evaluate the effectiveness of UBPD in terms of the two following rates:

- **False negative:** The system fails to recognise a phishing attack.
- **False positive:** The system recognises a legitimate website as a phishing website.

In addition we also search for a useful default threshold value (mentioned in Sect. 3.2). For both experiments UBPD was modified to not present the warning dialogue, instead it records the phishing score results as well as the URLs for later analysis.

5.1 False negative rate

From PhishTank[28] and millersmiles[21] we randomly collected 463 recently reported phishing webpages, which target Ebay, Paypal, and Natwest bank. We created four user profiles, which describe four artificial users' binding relationships with the three targeted websites. The four user profiles have different characteristics as shown in Table 1. 'Reuse' indicates maximum possible reuse of authentication credentials. In this case the user would have same user name and password for Ebay and Paypal. 'Uniqueness' indicates whether the user would use the exact data they shared with a legitimate website at other places. For example if Bob chooses his email address as password then the uniqueness is weak, because Bob is very likely to tell other websites his email address. If Bob uses some random string as his password, then the uniqueness is strong, because this random string is unlikely to be used with any other websites.

We entered the artificial authentication credentials to each phishing webpages. Regardless of the characteristics of the

Table 2 Statistics of phishing websites based on the type of information requested. AC: Authentication Credentials; PI: Personal Information

	Ebay	Paypal	Natwest
AC	211	176	39
AC + PI	22	5	6
PI	4	0	0
Total	237	181	45

user profile, the detection result is the same for all four users: 459 pages had a phishing score of 1, and 4 had a phishing score of 0. Thus only four pages evaded detection—a 99.14 percent detection rate. Compared to other existing phishing website detection systems, UBPD's detection rate may not be significantly better. **Its biggest advantage** is that its detection method detects essential characteristics of a phishing attack, namely that phishing web pages request authentication credentials. The details of how users may be manipulated may change with future phishing attacks, but the requesting of such details remains constant. Other detection systems based on the analysis of incoming data will need to adapt and be redesigned for future phishing attacks; UBPD will not.

Detailed analysis confirms that the detection result is determined mainly by the information requested by the phishing webpage. Table 2 shows the classification of the phishing webpages based on the type of information they requested. 92% of the collected phishing webpages asked only for authentication credentials and 7.14% of the collected phishing webpages asked both for personal and authentication credentials.

The four phishing web pages that UBPD failed to detect asked only for personal information such as full name, address, telephone number and mother's maiden name. In fact, they can not be detected by UBPD no matter what the threshold value is. However, it is impractical for phishing attacks to ask for personal information without requesting authentication credentials first, because those phishing webpages are not presented to users when a user victim first arrives at the phishing website. Those phishing websites would normally first present the user with a login webpage before directing the user to the webpage that asking for the personal information (none of the four phishing webpages were the landing page of the phishing attacks). Otherwise such practice would be seemed abnormal, and make potential victims very suspicious. As a result, UBPD can detect the phishing attacks and stop users from even reaching the phishing webpages that ask for personal information.

The sample size in this experiment is large and we might have some expectation that this would be reasonably indicative of success rate when deployed 'in the wild'.

5.2 False positive rate

Five volunteers were provided with the information needed to install UBPD on their machine. We did not explicitly ask them to train UBPD with all their binding relationships, because we wanted to see how users would train UBPD and what the false positives would be in reality if the user has not properly trained UBPD. At the end of one week, we collected the result log from their machines.

The volunteers were three male and two female science students. They all used Firefox as their main web browser. They were all regular Internet users (in average over three hours per day). As a result the UBPD was activated a large number of times and the interactions that occurred during the experiments covered a wide range of types of interaction. Another reason we chose those volunteers is because they are the most unlikely user group to fall victims to phishing attacks [13] and so we can safely assume they have all interacted with legitimate websites. In total the volunteers interacted with 76 distinct websites, submitted data to those websites 2107 times, and UBPD ran in detection mode only 81 times. In fact all the websites volunteers visited were legitimate. On 59 occasions the phishing score was 0, on five interactions gave a score of 0.25, on 13 occasions the score was 0.5, and the score was 1 on three occasions.

The phishing score was 1 when users interacted with three legitimate websites (the registration webpages of videojug.com and surveys.com, and the authentication webpage of a web forum). We asked the volunteers what data they supplied to those webpages. It seems that the reuse of authentication credentials on creating new accounts is the reason. In this experiment, the warning dialog is not presented, as we did not aim to test usability. But if it does, then the user must make a decision to train UBPD to remember these new binding relationships. To avoid the user's confusion about what is the right choice when the warning dialog is presented, the dialog always reminds the user of the legitimate websites UBPD is aware of, and tells the user that if the user is sure the current website is legitimate, and the website is not remembered by UBPD, then they need to update their binding relationships (see Fig. 4 in Sect. 3.2). This requires no technical knowledge and should be quite easy to understand. There are only two choices provided by the dialog: to update the profile and submit the data; or do not send the user submitted data and close the phishing webpage. There is no third choice provided by the dialog, in this way we force the user to make the security decision and they can not just ignore the warnings given by the system.

Many websites force users to supply an email address as the user name. As a result, the user's email address is kept in the user profile as part of user's authentication credentials. This email address almost inevitably will be given out to other websites, which are not contained in the user profile, for various reasons such as contact method, activate the

newly registered account, etc. Thus even when the user does not intend to give out their credentials, the email address nevertheless is shared and our system simply confirmed that by calculating the phishing score of 0.5 (which means half of the data the user has shared with a legitimate website was given away to a website that was not in user's profile) on 13 occasions.

For one volunteer five interactions gave a phishing score of 0.25. The user had an account at a major bank, the authentication credentials for which compromised four data items. One of these was the family name. For other sites not included in the user's profile asking for this information caused our system to identify the sharing of the data.

Based on the figures from both experiments we decided to set the default threshold value to 0.6. First, it can successfully detect phishing webpages asking for more than half of the credentials the user has shared with a legitimate website (99.14% of the phishing websites in Experiment One can be detected). It also generated few false positives. The false positive rate of the system is 0.0014 (obtained by dividing the number of false positives generated with the total number of times the UBPD was activated).

Of course this is only a preliminary false positive evaluation, but it still shows the system has a small false positive rate, and it also shows that the reuse of the authentication credentials and partial training are the main cause of the false positives.

6 Discussion

6.1 Why UBPD is useful

Besides its high detection accuracy, UBPD is useful also because it complements existing detection systems. First UBPD detects phishing websites based on users' behaviours, not the incoming data that attackers can manipulate freely. Violation of the binding relationships cannot be changed no matter what techniques phishers choose to use. As the evaluation proves, UBPD is able to consistently detect phishing webpages regardless of how they are implemented as long as they ask for authentication credentials. In contrast detection systems based on incoming data may find it difficult to deal with novel and sophisticated spoofing techniques. UBPD analyses the identity of websites using both IP addresses and domain names, it can detect pharming attacks, which are undetectable by many existing systems. Being independent of the incoming data means low cost in maintenance, the system does not need updating when attackers vary their techniques, and so we have far fewer evasion techniques to deal with.

Some systems that try to stop phishing attacks from reaching the users (phishing site take down, botnet take

down, phishing email filter, etc.), some have tried to detect phishing webpages as soon as users arrive at a new web page (Phishing URL blacklist, netcraft toolbar, spoofguard, CARTINA, etc.), and some have tried to provide useful information to help users to detect phishing websites. However, there are no other systems that work at the stage when phishing webpages have somehow penetrated through and users have started to give out information to them. Thus our system plugs a major remaining gap, and it can be easily combined with other phishing detection systems to provide multi-stage protection.

6.2 Performance

The two working modes of UBPD reduce the computing complexity. Computing in detection mode is more expensive, but it runs only occasionally (in our second experiment UBPD only in detection mode 81 out of 2107 times). Computing in the detection mode is still light weight for the computing power of an average personal computer. (None of the volunteers noticed any delay.) As a result, UBPD is efficient to run and adds little delay to the existing user-webpage interaction experience.

6.3 Limitations and future work

UBPD should be viewed as an initial but promising effort towards detecting phishing by analysing user behaviour. Despite its detection effectiveness, there are some limitations within the implementation and design.

Currently UBPD cannot handle all types of authentication credentials. It can handle static type authentication credentials such as user name, password, security questions, etc. but dynamic authentication credentials shared between users and the legitimate websites cannot be handled by the current implementation (e.g. one-time passwords). In future we will investigate how to handle such credentials.

There is also another method for a phisher to evade detection. Attackers might compromise a website within a user's personal whitelist and host the phishing website under the website's domain (perhaps for several hours to a couple of days). This method is difficult to carry out, and it is unlikely to happen if there are still easier ways to launch phishing attacks.

Detection itself requires little computing time, in fact, retrieving the data from WHOIS database is the most time consuming part. It depends on the type of network the user is using and the workload on the WHOIS database. In future we could add some cache functionality to reduce the number of queries for each detection. Currently the system produces a slight delay because of the WHOIS database query.

Although we have paid attention to the usability design of the system, we have not systematically evaluated it. Further user evaluation will inform development.

Each user profile contains useful security information, such as the websites the user has binding relationship with. Once deployed, UBPD is likely to discover the active phishing websites at the earliest stages. In future we also plan to investigate how this information can be collected safely and used to provide more accurate and timely detection.

7 Conclusion

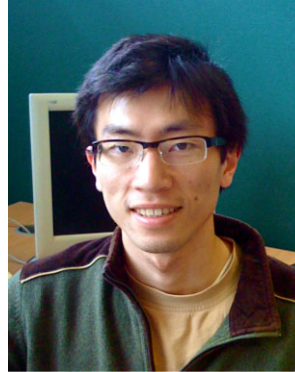
We have used a novel paradigm—analysis of the users' behaviours—to detect phishing websites. We have shown that it is an accurate method, discussed how it has been designed and implemented to be hard to circumvent, and have discussed its unique strength in protecting users from phishing threats. UBPD is not designed to replace existing techniques. Rather it should be used to complement other techniques, to provide better overall protection. We believe our approach fills a significant gap in current anti-phishing technology capability.

Acknowledgements We wish to thank all the people helped us, and especially the Engineering and Physical Sciences Research Council (EPSRC) of United Kingdom for their sponsorship of the project "Defending the weakest link: Intrusion via Social Engineering" (EPSRC Grant EP/D051819/1).

References

1. Abad, C. (2005). The economy of phishing: a survey of the operations of the phishing market. *First Monday*, 10(9).
2. Albrecht, K., Burri, N., & Wattenhofer, R. (2005). Spamato— an extendable spam filter system. In *2nd Conference on email and anti-spam (CEAS)*, Stanford University, Palo Alto, California, USA, July 2005.
3. Behera, P., & Agarwal, N. (2006). A confidence model for web browsing. In *Toward a more secure web—W3C workshop on transparency and usability of web authentication*, 2006.
4. Chandrasekaran, M., Chinchain, R., & Upadhyaya, S. (2006). Mimicking user response to prevent phishing attacks. In *IEEE international symposium on a world of wireless, mobile, and multi-media networks*, 2006.
5. Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., & Mitchell, J. C. (2004). Client-side defence against web-based identity theft. In *NDSS '04: proceedings of the 11th annual network and distributed system security symposium*, February 2004.
6. Dhamija, R., Tygar, D., & Hearst, M. (2006). Why phishing works. In *CHI '06: proceedings of the SIGCHI conference on human factors in computing systems*, ACM Special Interest Group on Computer-Human Interaction 2006 (pp. 581–590).
7. Fette, I., Sadeh, N., & Tomasic, A. (2007). Learning to detect phishing emails. In *WWW '07: proceedings of the 16th international conference on world wide web*, New York, NY, USA, 2007 (pp. 649–656). New York: ACM Press.
8. Florencio, D., & Herley, C. (2007). A large-scale study of web password habits. In *WWW '07: proceedings of the 16th international conference on world wide web*, New York, NY, USA, 2007 (pp. 657–666). New York: ACM Press.

9. Franco, R. (2005). Better website identification and extended validation certificates in ie7 and other browsers. In *IEBlog*, November 2005.
10. Goodger, B., Hickson, I., Hyatt, D., & Waterson, C. (2001). *Xml user interface language (xul) 1.0* (Technical report). Mozilla Org.
11. Hartman, S. (2007). *Ietf-draft: requirements for web authentication resistant to phishing* (Technical report). MIT.
12. Hhrmann, B., Htgaret, P. L., & Pixley, T. (2007). *Document object model events* (Technical report). W3C.
13. Jagatic, T., Johnson, N., Jakobsson, M., & Menczer, F. (2007). Social phishing. *ACM Communication*, October.
14. Jakobsson, M. (2005). Modeling and preventing phishing attacks. In *Phishing panel in financial cryptography '05*, 2005.
15. Jakobsson, M. (2007). Human factors in phishing. In *Privacy & security of consumer information '07*, 2007.
16. Jakobsson, M., Tsow, A., Shah, A., Blevis, E., & Lim, Y.-K. (2007). What instills trust? A qualitative study of phishing. In *Extended abstract, USEC '07*, 2007.
17. Johnston, P. A. (2009). <http://pajhome.org.uk/crypt/index.html>.
18. Litan, A. (2006). *Toolkit: E-commerce loses big because of security concerns* (Technical report). Gartner Research.
19. McCall, T. (2007). *Gartner survey shows phishing attacks escalated in 2007; more than \$3 billion lost to these attacks* (Technical report). Gartner Research.
20. Microsoft (2005). *Anti-phishing white paper* (Technical report). Microsoft.
21. MillerSmiles (2009). Official website. <http://www.millersmiles.co.uk>.
22. Moore, T., & Clayton, R. (2007). Examining the impact of website take-down on phishing. In *eCrime '07: proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, New York, NY, USA, 2007 (pp. 1–13). New York: ACM Press.
23. Mozilla (2007). Phishing protection. <http://www.mozilla.com/en-US/firefox/phishing-protection/>.
24. Netcraft (2007). <http://toolbar.netcraft.com/>.
25. Ollmann, G. (2005). *The pharming guide* (Technical report). Next Generation Security Software Ltd.
26. Ollmann, G. (2009). *The phishing guide* (Technical report). NGSS.
27. Pan, Y., & Ding, X. (2006). Anomaly based web phishing page detection. *Acsac*, 0, 381–392.
28. Phishtank (2007). <http://www.phishtank.com/>.
29. Schechter, S., Dhamija, R., Ozment, A., & Fischer, I. (2007). The emperor's new security indicators: an evaluation of website authentication and the effect of role playing on usability studies. In *2007 IEEE symposium on security and privacy*, 2007.
30. Security, R. (2007). *Enhancing one-time passwords for protection against real-time phishing attacks* (Technical report). RSA.
31. Staikos, G. (2005). Web browser developers work together on security. Web, November.
32. van Kesteren, A. (2006). *The xmlhttprequest object* (Technical report). W3C.
33. W3C. (2006). Web security context—working group charter. Web.
34. Watson, D., Holz, T., & Mueller, S. (2005). *Know your enemy: phishing* (Technical report). The HoneyNet Project & Research Alliance.
35. Wu, M., Miller, R. C., & Garfinkel, S. L. (2006). Do security toolbars actually prevent phishing attacks? In *CHI '06: proceedings of the SIGCHI conference on human factors in computing systems*, New York, NY, USA, 2006 (pp. 601–610). New York: ACM Press.
36. Wu, M., Miller, R. C., & Little, G. (2006). Web wallet: preventing phishing attacks by revealing user intentions 2006 (pp. 102–113).
37. Zhang, Y., Hong, J. I., & Cranor, L. F. (2007). Cantina: a content-based approach to detecting phishing web sites. In *WWW '07: proceedings of the 16th international conference on world wide web*, New York, NY, USA, 2007 (pp. 639–648). New York: ACM Press.



Xun Dong is studying the Ph.D. degree in the computer science department of university of York since 2006 under the supervision of Professor John A. Clark and Dr. Jeremy Jacob. His early work focused on intrusion detection and since 2007 he started to work on a research project on protecting users from the threats of phishing attacks. He published research papers regards the human factors in phishing attacks and threat modeling techniques when human users are involved.



John A. Clark He left Oxford in 1987 to join the software and systems house Logica. He spent five years working under contract for HMG on matters of security evaluation and security R&D. He joined York in 1992. His research focused on cryptography or security related things largely from a non-standard computational perspective. He has also led the software testing research work in York. He is author or coauthor of around 80 publications since 1998.



Jeremy L. Jacob has a D.Phil. and M.Sc. (Computation) from the University of Oxford, and a B.Sc. (Mathematics) from the University of Hull. He now works at the University of York, where he pursues his interests in application of mathematics to computation and security.