On the Effectiveness of Techniques to Detect Phishing Sites

Christian Ludl, Sean McAllister, Engin Kirda, Christopher Kruegel

Secure Systems Lab, Technical University Vienna
[chris2,sean,ek,chris]@seclab.tuwien.ac.at

Abstract. Phishing is an electronic online identity theft in which the attackers use a combination of social engineering and web site spoofing techniques to trick a user into revealing confidential information. This information is typically used to make an illegal economic profit (e.g., by online banking transactions, purchase of goods using stolen credentials, etc.). Although simple, phishing attacks are remarkably effective. As a result, the numbers of successful phishing attacks have been continuously increasing and many anti-phishing solutions have been proposed. One popular and widely-deployed solution is the integration of blacklist-based anti-phishing techniques into browsers. However, it is currently unclear how effective such blacklisting approaches are in mitigating phishing attacks in real-life. In this paper, we report our findings on analyzing the effectiveness of two popular anti-phishing solutions. Over a period of three weeks, we automatically tested the effectiveness of the blacklists maintained by Google and Microsoft with 10,000 phishing URLs. Furthermore, by analyzing a large number of phishing pages, we explored the existence of page properties that can be used to identify phishing pages.

1 Introduction

Online services simplify our lives. They allow us to access information ubiquitously and are also useful for service providers because they reduce the operational costs involved in offering a service. For example, online banking over the web has become indispensable for customers as well as for banks. Unfortunately, interacting with an online service such as a banking web application often requires a certain degree of technical sophistication that not all Internet users possess. For the last couple of years, such naive users have been increasingly targeted by phishing attacks that are launched by miscreants who are aiming to make an easy profit by means of illegal financial transactions. Phishing is a form of electronic identity theft in which a combination of social engineering and web site spoofing techniques are used to trick a user into revealing confidential information with economic value. In a typical attack, the attacker sends a large number of spoofed (i.e., fake) e-mails to random Internet users that appear to be coming from a legitimate business organization such as a bank. The e-mail urges the recipient (i.e., the potential victim) to update his personal information. Often, the e-mail also warns the recipient that the failure to comply with

the request will result in the suspending of his online banking account. Such ungrounded threats are common in social engineering attacks and are an effective technique in persuading users.

When the unsuspecting victim follows the phishing link that is provided in the e-mail, he is directed to a web site that is under the control of the attacker. The site is prepared in a way such that it looks familiar to the victim. That is, the phishers typically imitate the visual corporate identity of the target organization by using similar colors, icons, logos and textual descriptions. In order to "update" his personal information, the victim is asked to enter his online banking login credentials (i.e., user name and password) to access the web site. If a victim enters his valid login credentials into the fraudulent web site, the phisher can then impersonate the victim. This may allow the attacker to transfer funds from the victim's account or cause other damage. Because victims are directly interacting with a web site that they believe they know and trust, the success rate of such attacks is very high. Note that although phishing has been receiving wide media coverage (hence, causing the number of Internet users who have heard of phishing to increase), such attacks still remain effective as phishers have been adapting their social engineering attempts accordingly. For example, many phishing e-mails now ask the victims to validate their personal information for "security purposes", supposedly because the targeted organization would like to protect them against the phishing threat.

According to the Anti-Phishing Working Group [2], the phishing problem has grown significantly over the last years. For example, the number of unique phishing web sites has exploded from 7,197 in December 2005 to 28,531 in December 2006. Also, financial losses stemming from phishing attacks have risen considerably, to more than $2.8 billion in the last year according to Gartner Inc. [8].

The phishing problem has become so serious that large IT companies such as Microsoft, Google, AOL and Opera have recently started announcing browser-integrated, blacklist-based anti-phishing solutions. However, one important question that still remains is how effective such blacklist-based solutions are in dealing with the phishing problem.

In this paper, we report our findings on analyzing the effectiveness of two popular blacklist-based anti-phishing solutions. We automatically tested the effectiveness of the blacklists maintained by Google and Microsoft over a three week period. During this time, we tested the blacklists with 10,000 phishing URLs. Furthermore, by analyzing a large number of phishing pages, we experimentally explored the existence of page properties that can be used to identify phishing pages.
The contributions of this paper are as follows:

- We show that blacklist-based solutions are actually quite effective in protecting users against phishing attempts. In our experiments, Google recognized almost 90% of the malicious URLs at the time of the initial check.
- By analyzing a large number of phishing pages, we built a classification model that attempts to use the properties of a page (e.g., number of password

fields, number of external links, etc.) to distinguish between malicious and legitimate pages. We believe our model can be used to improve existing anti-phishing approaches (e.g., such as the built-in phishing detection heuristics used by IE 7).

2 Related Work

A number of anti-phishing solutions have been proposed to date. Some approaches attempt to solve the phishing problem at the e-mail level. That is, they try to prevent phishing e-mails from reaching the potential victims by means of filters and content analysis. Obviously, such techniques are closely related to anti-spam research. In fact, anti-spam techniques (e.g., such as Bayesian filters) have proven to be quite effective in also intercepting phishing e-mails. Unfortunately, the effectiveness of anti-spam techniques often depends on many critical factors such as regular filter training and the availability of anti-spam tools. Furthermore, filtering, no matter how efficient, is not perfect and some phishing e-mails may manage to get through the filters and reach potential victims (i.e., strengthening the belief that the e-mail is legitimate).

Microsoft and Yahoo have also defined e-mail authentication protocols (i.e., Sender ID [16] and DomainKeys [32]) that can be use to verify if a received e-mail is authentic. The main disadvantage of these solutions, however, is that they are currently not used by the majority of Internet users.

Several academic, browser-integrated solutions (i.e., client-side techniques) have been proposed to date to mitigate phishing attacks. Well-known solutions in literature are SpoofGuard [3, 26] and PwdHash [24, 23]. SpoofGuard looks for phishing symptoms (e.g., obfuscated URLS) in web pages and raises alerts. PwdHash, in contrast, creates domain-specific passwords that are rendered useless if they are submitted to another domain (e.g., a password for `www.gmail.com` will be different if submitted to `www.attacker.com`). Our anti-phishing tool, AntiPhish [11] takes a different approach and keeps track of where sensitive information is being submitted. That is, if it detects that confidential information such as a password is being entered into a form on an untrusted web site, a warning is generated and the pending operation is canceled.

An interesting solution that has been proposed by Dhamija et al. [5] involves the use of a so-called dynamic security skin on the user's browser. The technique allows a remote server to prove its identity in a way that is easy for humans to verify, but difficult for phishers to spoof. The disadvantage of this approach is that it requires effort by the user. That is, the user needs to be aware of the phishing threat and check for signs that the site he is visiting is spoofed. In fact, in a later study [6], Dhamija et al. report that more than 20% of the users do not take visual cues into consideration when surfing and that visual deception attacks can fool even the most sophisticated users.

Lui et al. [30] analyze and compare legitimate and phishing web pages to define metrics that can be used to detect a phishing page. A web page is classified as a phishing page if its visual similarity value is above a pre-defined threshold.

The most popular and widely-deployed techniques, however, are based on the use of blacklists of phishing domains that the browser refuses to visit. For example, Microsoft has recently integrated a blacklist-based anti-phishing solution into its Internet Explorer (IE) 7 browser. The browser queries lists of blacklisted and whitelisted domains from Microsoft servers and makes sure that the user is not accessing any phishing sites. Microsoft's solution is also known to use some heuristics to detect phishing symptoms in web pages [27, 15]. Obviously, to date, the company has not released any detailed public information on how its anti-phishing techniques function.

Other browser-integrated anti-phishing tools include Google Safe Browsing [25], NetCraft tool bar [18], eBay tool bar [7] and McAfee SiteAdvisor [12]. Similar to the Microsoft IE 7 anti-phishing protection, Google Safe Browsing uses blacklists of phishing URLs to identify phishing sites. The disadvantage of the approach is that non blacklisted phishing sites are not recognized. In contrast, NetCraft assesses the phishing probability of a visited site by trying to determine how old the registered domain is. The approach partially uses a database of sites that are maintained by the company. The downside of the approach, hence, is that new phishing sites that are not in the database might not be recognized. Similarly, SiteAdvisor is a database-backed solution that is, however, mainly designed for protection against malware-based attacks (e.g., spyware, Trojan horses, etc.). It includes automated crawlers that browse web sites, perform tests and create threat ratings for each visited site. Unfortunately, just like other blacklist or database-based solutions, SiteAdvisor cannot recognize new threats that are unknown and not in the database. The eBay solution is specifically designed for eBay and PayPal and involves the use of a so-called "Account Guard" that changes color if the user is on a spoofed site.

Verisign has also been providing a commercial anti-phishing service [28]. The company is crawling millions of web pages to identify "clones" in order to detect phishing web sites. Furthermore, just like other large companies such as Microsoft, McAfee and Google, blacklists of phishing web sites are maintained.

Note that one problem with crawling and blacklists proposals could be that the anti-phishing organizations will find themselves in a race against the attackers. This problem is analogous to the problems faced by anti-virus and anti-spam companies. Obviously, there is always a window of vulnerability during which users are susceptible to attacks. Furthermore, listing approaches are only as effective as the quality of the lists that are maintained. Hence, one interesting research question is how effective such blacklists are in mitigating attacks.

In late 2006, two studies appeared that compared the effectiveness of the Google and Microsoft blacklists. One study, which was paid for by Microsoft, unsurprisingly concluded that the Microsoft blacklist is superior [19]. The other study, initiated by Mozilla, drew the opposite conclusion [17]. Thus, we felt that a third, independent evaluation would be valuable. In addition, the two studies mentioned above only consider whether a phishing URL was blacklisted at one point in time. However, no attempt was made to assess whether phishing URLs

were added at a later time, or whether they were never added at all. Hence, a key difference of our study is that we take these questions into account.

Also, independently and concurrently from our work, Zhang et al. [33] have also performed a similar study that investigates the efficiency of anti-phishing solutions. The authors have created an automated test-bed with which they have tested the detection rates of mostly blacklist-based anti-phishing solutions. An important difference of our work is that our tests and experimental data include 10,000 phishing URLs collected over a three week period lasting from December 2006 to January 2007. In comparison, Zhang et al.'s dataset includes 100 phishing URLs collected over a period of three days in November 2006. Furthermore, in our work, besides investigating the efficiency of popular blacklists, we also experimentally explored the existence of page properties that can be used to identify phishing pages.

3 Scope of Study

The goal of this paper is to analyze the effectiveness of anti-phishing solutions. More precisely, we are interested in assessing techniques that are capable of classifying individual web pages. To qualify for our study, a technique must be capable of determining whether a page is legitimate or a phishing page, given only the URL and the page's source code. We did not consider mechanisms that aim to prevent users from visiting a phishing site (e.g., by recognizing phishing mails). Also, we did not evaluate solutions that attempt to protect sensitive user information from being leaked to the phishers (e.g., by replacing passwords with site-specific tokens, or by using novel authentication mechanisms). Currently, there are two main approaches to classify visited web pages without any additional information. The first one is based on URL blacklists. The second approach analyzes properties of the page and (sometimes) the URL to identify indications for phishing pages.

Blacklists: Blacklists hold URLs (or parts thereof) that refer to sites that are considered malicious. Whenever a browser loads a page, it queries the blacklist to determine whether the currently visited URL is on this list. If so, appropriate countermeasures can be taken. Otherwise, the page is considered legitimate. The blacklist can be stored locally at the client or hosted at a central server.

Obviously, an important factor for the effectiveness of a blacklist is its coverage. The coverage indicates how many phishing pages on the Internet are included in the list. Another factor is the quality of the list. The quality indicates how many non-phishing sites are incorrectly included into the list. For each incorrect entry, the user experiences a false warning when she visits a legitimate site, undermining her trust in the usefulness and correctness of the solution. Finally, the last factor that determines the effectiveness of a blacklist-based solution is the time it takes until a phishing site is included. This is because many phishing pages are short-lived and most of the damage is done in the time span

between going online and vanishing. Even when a blacklist contains many entries, it is not effective when it takes too long until new information is included or reaches the clients.

For our study, we attempted to measure the effectiveness of popular blacklists. In particular, we studied the blacklists maintained by Microsoft and Google. We believe that these blacklists are the ones that are most wide-spread, as they are used by Internet Explorer and Mozilla Firefox, respectively.

Page analysis: Page analysis techniques examine properties of the web page and the URL to distinguish between phishing and legitimate sites. Page properties are typically derived from the page's HTML source. Examples of properties are the number of password fields, the number of links, or the number of unencrypted password fields (these are properties used by SpoofGuard [3]).

The effectiveness of page analysis approaches to identify phishing pages fundamentally depends on whether page properties exist that allow to distinguish between phishing and legitimate sites. Thus, for our study, we aimed to determine whether these properties exist, and if so, why they might be reasonable candidates to detect phishing pages.

In a first step, we defined a large number of page properties that can be extracted from the page's HTML code and the URL of the site. Then, we analyzed a set of phishing and legitimate pages, assigning concrete values to the properties for each page. Finally, using the collected data as training input, we applied machine-learning techniques to create a web page classifier. The resulting classifier is able to distinguish well between phishing and legitimate classifiers, with a very low false positive rate. This indicates that the aforementioned page properties that allow one to identify malicious pages do in deed exist, at least for current phishing pages.

It seems that Microsoft has drawn a similar conclusion, as the new Internet Explorer browser also features a phishing page detection component based on page properties. This component is invoked as a second line of defense when a blacklist query returns no positive result for a visited URL. As part of our study, we attempted to determine the features that are most relevant to the IE for identifying phishing sites. We observed that the IE model looks different than the one we have built, and also detects less phishing pages.

4 Experimental Setup

In this section, we first discuss the anti-phishing solutions that we chose to examine. Then, we describe the test data that was collected to conduct our experiments.

4.1 Anti-phishing Solutions

In the previous section, we outlined the scope of our study. In particular, we explained our focus on solutions that analyze web pages for indications of phishing,

namely blacklist-based and page analysis techniques. To evaluate the effectiveness of these approaches, it is desirable to select solutions that are in wide-spread use. This is important so that the results of the study are relevant. Also, a wide-spread solution has a higher likelihood of being well-supported and maintained, thus making the result of the study meaningful. Consider a study that evaluates the effectiveness of a blacklist that is not updated. While this study will probably conclude that blacklists are very ineffective, these results are not very insightful.

For this study, we decided to analyze the effectiveness of the anti-phishing solutions used by the Microsoft Internet Explorer 7 and Mozilla Firefox 2. The reasons for this choice are the following: First, these two applications are the most-used web browsers on the Internet. Second, both browsers recently introduced anti-phishing mechanisms, and one can assume that these mechanisms will be the most widely deployed anti-phishing solutions in the near future. Note that we did not include the Opera browser in our study because the company announced a phishing filter only shortly after we started our experiments.

Internet Explorer 7: Microsoft recently introduced the version 7 of its popular Internet Explorer (IE) browser, which was automatically deployed to millions of computers around the world via Microsoft's Windows update web site. One of the most important, new features of this browser is its anti-phishing support. To this end, the IE 7 uses both an online database of reported phishing sites as well as heuristics that analyze web pages to determine the potential risk of a web site [27, 15]. This makes the Internet Explorer an optimal selection for our study, as it uses both a blacklist and page analysis approaches to identify phishing attempts.

For the user, the anti-phishing support has the following visible effects: If a site is a reported phishing site, the address bar of the browser turns red and the web site turns into a full page warning about the potential dangers of the site. The user can then choose to either proceed to the site or close the page. If the site is not found in the blacklist of reported scam pages, but the page heuristics detect a possible phishing attempt, the address bar turns yellow and a warning symbol appears at the bottom of the screen.

Mozilla Firefox: Mozilla Firefox is considered the only serious competitor to the Microsoft IE, which currently dominates the browser market. Since version 2.0, Firefox includes anti-phishing support. The browser can connect to any available blacklist provider, using a documented, open protocol. Currently, however, only the Google blacklist servers[1] are pre-configured. The anti-phishing approach of Firefox is solely based on blacklists and does not use any form of page analysis to warn the user of potential scams. By default, Firefox uses regularly downloaded lists and does not perform blacklist lookups with each web connection. The user, however, can also choose to use live blacklists. When a visited URL is on the blacklist, Firefox turns the web page into a black-and-white version and displays a warning message.

[1] Google's blacklist is also used by the Google safe browsing toolbar [9].

4.2 Test Data

For our study, a large number of phishing pages were necessary. We chose `phishtank.com` as a source of phishing URLs. The information from this site is freely available and the amount of reported phishing sites is very large (approximately five hundred new phishing reports every day). There are other providers of blacklist data, but their feeds are typically only available for a fee, and their reports tend to focus on particular phishing incidents [4] (for example, a large scale phishing attack towards a particular institution, with multiple site mirrors and copies of spoofed emails), whereas the phishtank datasets focus on the URL itself, thereby making it more appropriate to our goals. The free availability of this information is also a very important aspect to our research, as phishtank is neither affiliated with Microsoft nor with Mozilla, making the results more objective. The URLs were extracted from a XML feed [20] of verified and (then) online phishing sites. In addition to saving the URLs we made local copies of each site. This was important, as most phishing sites are only online for a short period of time and the page source was needed later for evaluating the effectiveness of page analysis techniques. Note that even when attempting to download phishing sites immediately, a significant fraction of these sites was already down (and thus, no longer available for further analysis).

Table 1. (a) Domains that host phishing sites. (b) Popular phishing targets.

(a)

No domain (numerical)	3,864
.com	1,286
.biz	1,164
.net	469
.info	432
.ws	309
.jp	307
.bz	256
.nz	228
.org	156
.de	111
.ru	106
.us	105

(b)

paypal	1,301
53.com	940
ebay	807
bankofamerica	581
barclays	514
volksbank	471
sparkasse	273
openplan	182
Total	5,069

Note that phishtank is a community-driven site that lives from submissions made by its users. Hence, this approach has the disadvantage that some reported sites may not be phishing sites. Phishtank uses a system based on verification by other users, who can vote whether a page is a phishing page or not. These votes are weighted by the experience and the rank the user has within the community. Nevertheless, it is possible that even a page that is verified by phishtank users

to be malicious is in fact legitimate. Therefore, we cannot completely rule out the possibility that some samples are false positives. Also, note that we were not able to investigate how often phishes reported by Google and Microsoft appeared on phishtank. This is because we do not have access to the full blacklist used at Microsoft (which is queried via SOAP requests) and because we believe that the Google blacklist that is available online is not complete (i.e., we suspect that the Google blacklist does not include IPs that have been taken down or that are not relevant anymore).

We started the collection of phishing pages on 15. December 2006, with the goal of gathering 10,000 URLs. This goal was reached after about three weeks, on 4. January 2007. During this time a webcrawler periodically checked the collected URLs and when possible downloaded the sites to our local repository. If the download failed the crawler checked the site again on its next run and continued doing so for 48 hours after the site was first added to our database. For the 10,000 URLs that we collected, we were only able to download the page sources of 4,633 sites. This clearly underlines the short time span that many phishing pages are online. In Table 1(a), the leading top level domain names of the phishing sites are listed. Note that the largest fraction of pages is not hosted under any domain, but uses only numerical IP addresses. Among the remaining sites, a large variety of domains can be observed.

We also analyzed the sites that were spoofed most often by checking for hostnames of legitimate sites in the phishing sites URL. This is also a property we checked for in the page analysis (see section 6.1). The results are shown in Table 1(b). Not surprisingly, both `paypal` and `ebay` were among the top three. Most other frequently targeted victims are online portals of banks. The eight most targeted victims alone account for more than 50% of the phishing URLs that we observed. Further analysis of current phishing targets based on Google's blacklist can be found at [13].

5 Study of Blacklist Effectiveness

To study the effectiveness of the blacklists provided by Microsoft (used by the Internet Explorer) and Google (used by Firefox), we periodically checked whether our most recently collected URLs from `phishtank.com` were already blacklisted. Depending on the initial response, we either saved the positive answer (i.e., the site is blacklisted) and stopped to check the URL, or we continued to send requests with not-yet blacklisted URLs. In the latter case, we stopped after the first positive response and saved the received data together with a timestamp.

Automated analysis: Because of the large amount of data that needed to be processed, an automated solution to check URLs was necessary. Hence, we had to exchange data with the two blacklist servers directly, without going through the browser. This task was quite easy for the Google blacklist server because the blacklist protocol and the specifications of the request and response formats are public [25, 21].

Of course, the situation is different for Microsoft's blacklist server, for which no public protocol information is available. The first problem that we faced was that the information between the IE and Microsoft is exchanged over an encrypted SSL connection. To discover more details about the protocol, we first set up an Apache SSL server. Then, we created a self-signed certificate and stored this certificate in our Internet Explorer. In addition, we added an entry for Microsoft's blacklist server `urs.microsoft.com` to the `hosts` file of the machine that the Internet Explorer was running on. The idea was to let the Microsoft blacklist server URL point to our SSL server. As a result, the browser contacted our server for each blacklist request. At this point, we were able to decrypt the messages that were arriving at our server, and we discovered that the communication was implemented via SOAP messages. We could then forward these messages to the real Microsoft server, and received responses that we could further analyze.

Analyzing the blacklist protocol, we observed that the URL of each visited page is stripped from any GET data at the end of the URL and then sent to the server for checking. This is different from the Google protocol, which always includes the complete URL. On sites that include iframes, a single request was sent for each iframe. This can lead to performance problems and was subsequently changed by Microsoft [29]. That is, before the change was introduced, an HTML page with an embedded iframe that linked to a known phishing site was reported as being a phishing site itself. After the change, we observed that the browser ignored iframe links. Hence, if we created a "phishing" page that used iframes and linked to a known phishing site, our phishing page would not be detected. During our experiments, we also noticed that the request and response pairs to the blacklist server often included lookup strings and responses for domains such as `microsoft.com`. Unfortunately, we were not able to determine the reason for these lookups. However, we could confirm that the browser version and the IP address of the computer that the browser is running on is sent to the Microsoft servers with each request (which is indicated in the privacy statement in [14]).

Analysis results: When checking the results of our test run, we recognized that for the 10,000 different URLs that we sent to both servers, only Google returned appropriate answers for all of them. However, we received only 6,131 responses from the Microsoft server. After further analysis of the data, we had to draw the conclusion that the Microsoft server was using some sort of rate limiting and locked us out because of too many requests. As a result, we unfortunately have less comprehensive data from the Microsoft blacklist. Nevertheless, we believe that we still have sufficient data to provide meaningful statistics for the Microsoft blacklist.

The results of the experiment are shown in Table 2. Initially, when we first requested the status for a URL, Google had 6,414 URLs on its blacklist. During the remaining time, 108 additional URLs were added to the blacklist. Thus, Google had blacklisted a total of 6,522 URLs (out of the 10,000 analyzed) at the end of our experiment. Microsoft sent a positive result for 3,095 URLs initially, and 331 later, which yields a total of 3,426 blacklisted pages. Because of the

different absolute numbers of checked URLs, the table also shows the relative values. Given our results, we observe that Google's blacklist appears to have more coverage, although the fraction of malicious pages that are detected are not too encouraging in both cases.

	Google	Microsoft
Sites	10,000 (100.00%)	6,131 (100.00%)
BL initially	6,414 (64.14%)	3,095 (50.48%)
BL delayed	128 (1.28%)	331 (5.40%)
BL total	6,522 (65.22%)	3,426 (55.88%)

Table 2. Blacklist responses for phishing URLs.

One problem with the results above is that they do not differentiate between phishing sites that are online (and thus, present a danger for users) and those that are offline. Many phishing sites have a very short lifetime, and as described in Section 4.2, we were only able to download the source for 4,633 of the 10,000 URLs we collected. Of course, maintainers of blacklists cannot include a URL when they cannot check the page that the URL is referring to. Thus, it is fairer to check the blacklist responses only for those pages that could be accessed. To this end, for the next experiment, we only considered those URLs for which both the Microsoft server and the Google server returned a response, and which could be successfully downloaded. For this, the data set contained a total of 3,592 URLs. The results are shown in Table 3. Interestingly, the hit rate is significantly higher in this case, suggesting that there are probably many URLs reported that are never considered for blacklisting because they are offline. Also, one can be seen that the gap between Google's blacklist and the one by Microsoft has increased, showing that Google delivers significantly better results.

	Google	Microsoft
Sites	3,595 (100.00%)	3,592 (100.00%)
BL initially	3,157 (87.89%)	2,139 (59.55%)
BL delayed	84 (2.34%)	274 (7.63%)
BL total	3,241 (90.23%)	2,413 (67.18%)

Table 3. Blacklist responses for live phishing sites.

For the next experiment, we analyzed the response times for URLs that were not initially blacklisted, but were entered into the list after some delay. More precisely, we considered all URLs that have not initially been blacklisted by a server, and measured the time until we first received a positive blacklist response. The fasted addition to the Microsoft blacklist occurred after 9:07 minutes, while

it took 9 days and almost 6 hours for the slowest. Google's fastest addition took 19:56 minutes, the slowest 11 days and 20 hours. On average, it took Microsoft 6.4 hours to add an initially not blacklisted entry (with a standard deviation of 6.2 hours). For Google, it took somewhat longer (on average 9.3 hours, with a standard deviation of 7.2 hours). Note that due to our test setup, we are not able to precisely measure the shortest listing times. This is because we do not continuously check URLs, but perform lookups periodically every 20 minutes. The shortest amount of time that passed between receiving the URL from phishtank and checking it for the first time was 7:33 minutes (for both servers). The longest period until checking Google's blacklist were 1:43 hours, and 2:10 for Microsoft's (due to unexpected problems with our scripts). In general, the results show that adding new entries to the blacklist often takes a considerable amount of time. However, only few entries were added overall, and the responses received from a server for a URL rarely changed over time.

Discussion: Looking at our results for two widely-used blacklists, we can conclude that this approach is quite successful in protecting users, especially when considering only URLs that refer to sites that are online. Especially Google, which has correctly recognized almost 90% of the malicious URLs at the time of the initial check, appears to be an important and powerful component in the fight against phishing.

Finally, it is worth pointing out that blacklist approaches may sometimes be defeated by simple obfuscation tricks, as reported in [1]. The basic idea of this attack is to replace single slashes with double slashes in phishing URLs, thereby defeating a simple blacklist string comparison. Both Firefox and Internet Explorer 7 were vulnerable to this kind of evasion.

6 Study of Page Analysis Effectiveness

To study the effectiveness of page analysis techniques, we first wish to answer the more basic question of whether page properties actually exist that allow one to distinguish between malicious (phishing) pages and legitimate (benign) ones. To answer this basic question, we define a number of properties that can be extracted from the source and the URL of web pages (described in Section 6.1). Once we define the page properties that we are interested in, we extract them from a set of phishing and normal (legitimate) pages. Based on the extracted properties, we use machine learning techniques to attempt to build a model to distinguish between malicious and legitimate pages. When such a model can be built, this implies that the properties must reflect some difference between phishing pages and normal ones. However, when such a classification model cannot be built, we have to conclude that the properties that we have defined do not allow to distinguish between phishing and legitimate pages. Our efforts of classifying web pages are discussed in Section 6.2. Finally, in Section 6.3, we use our properties to analyze the effectiveness of the page analysis heuristics implemented by Microsoft Internet Explorer 7.

6.1 Page Properties

As mentioned previously, we need to define appropriate properties to characterize a web page before it can be analyzed for indications that might reveal it as a phishing site. Not all of these properties have the same significance towards the probability of being a phishing site, but those that do not matter are then considered irrelevant by the data mining tool and therefore not included in the final decision tree. The following is the list of 18 properties that we consider. Our features are mostly extracted from the HTML source of a page. Two features are derived from the page's URL.

- **Forms:** Phishing pages aim to trick users into providing sensitive information. This information is typically entered into web forms. Thus, the *number of forms* (which is counted by this property) might provide an indicator to distinguish between phishing and legitimate pages, because some phishing sites ask the user to enter more than just his username and password (TAN numbers in banking applications and similar).
- **Input fields:** Because of the importance and prevalence of web forms on phishing pages, we aimed to define additional properties that characterize their structure in more detail. We specified properties that count the number of *input fields*, *text fields*, *password fields*, *hidden fields*, and *other fields*. The category *other fields* summarizes all input elements that are not member of any of the four more specific classes. Examples for *other fields* are radio buttons or check boxes.
- **Links:** Another important, general characteristic of every web page is its link structure. This not only takes into account links to other web pages, but also includes links to embedded images. Interestingly, many phishing pages contain links to the site they spoof, often to include original page elements from the victim page. To recognize such pages, we include properties that count the number of *internal links* to resources located in the page's domain as well as *external links* to resources stored on other sites. These links are extracted from a page by looking for `<a>` tags in the HTML source. By scanning for `<img>` tags, we extract links to *internal images* and *external images*. In addition, there is a category called *other links*, which counts the number of links included by other HTML tags (such as links to style sheets or JavaScript code, using the `<link>` tag). Furthermore, we explicitly count the number of (internal and external) links over a secure connection (i.e., by specifying an https target), using the *secure links* property. The same is done for images (*secure images*). Finally, to underline the importance of external links for finding phishing pages, we also define the category *external references*, which holds the sum of the number of *external links* and *external images*.
- **Whitelist references:** As mentioned previously, phishing pages often contain references to resources on their victims' sites. This fact is partly captured by the properties that count the number of external references. However, we can go one step further and analyze all external references for the presence

of links that are particularly suspicious. An external reference is suspicious when it points to a resource on a site that is a frequent target of phishing. To find such links, we check whether any of the links on a page refer to a resource on a trusted site. Trusted sites are those that appear on a whitelist. More precisely, we used a whitelist compiled by Google [10] that at the time of writing (February 2007) contained 2,431 entries that were considered trusted. This whitelist is freely available, in contrast to a similar whitelist maintained by Microsoft, which is stored in encrypted form in the Windows registry.

- **Script tags:** To distinguish between sites that make ample use of JavaScript and plain text pages, we count the number of JavaScript tags on a page and store it in the *script tags* property.
- **Suspicious URL:** This and the following property are derived from the URL of the page that is analyzed, and not from the page's source.
  An important goal of phishers is to make the phishing page appear as similar as possible to the spoofed one. Phishers often include parts of the URL they spoof into the URL of their phishing pages (for example, as part of the hostname, or in the path field). To capture such behavior, we search the URL for appearances of fragments of trusted pages. More precisely, we make use of the domains stored in Google's whitelist, and we check whether any of the trusted domains appear in the URL of the page that is currently analyzed. Hence, we perform a simple string search and determine whether any of the domains on the whitelist appear as a substring in the current URL. For example, `www.ebay.com` is on Google's whitelist. To check whether the current URL is suspicious, we check it for the presence of the substring `ebay`. Unfortunately, this approach can raise false positives, especially when trusted domains are very short. To mitigate this problem, we decided to only check for the appearance of domain names that have five or more characters. In addition, we manually added a few shorter domain names (such as `ebay` or `dell`) that are known to be frequently targeted by phishers. The remaining whitelist then contained 1,830 domains.
- **Uses SSL:** Another characteristic that was analyzed for each page is whether it is accessed over SSL (https) or not. In our preliminary studies, we observed that not many phishing sites make use of a secure server. One explanation could be that it is not straightforward to obtain a trustworthy certificate. Hence, not having such a certificate causes the browser to display a warning message, thereby alerting the user.

Of course, we are aware of the fact that our properties might not be complete. Furthermore, determined attackers could evade defense systems based on these properties (for example, by making use of Flash). However, the aim of our study is to understand whether current phishing pages can be identified based on page properties. Thus, we believe that our selection is reasonable and reflects the structure and methods that phishers use today. Also, our property list covers most page attributes that are checked by SpoofGuard [3], a tool that analyzes pages for phishing indicators. One property that is used by SpoofGuard, but

that we have not included are checks for techniques that attempt to obfuscate links. These techniques are already handled (checked) by browsers, which raise appropriate warning (Firefox 2) or error messages (Internet Explorer 7). As a result, they are no longer effective and, as a consequence, no longer used by phishers.

6.2 Classification Model

Based on the properties defined in the previous section, we built a classification model that attempts to use these properties to distinguish between malicious and legitimate pages.

As the set of phishing sites, we used the 4,633 pages that were successfully downloaded during our experiments (as discussed in Section 4.2) plus about 1100 pages that we collected before starting the blacklist analysis, resulting in 5751 analyzed sites. To obtain a set of legitimate pages, we had to collect a reasonable amount of comparable benign sites. Since the targets of phishers are mostly login pages, we used Google's `inurl:` operator to search for login pages. More precisely, we used Google to search for pages where one of the following strings `login`, `logon`, `signin`, `signon`, `login.asp`, `login.php` and `login.htm` appears in the URL. After downloading 5,124 pages, we manually removed from our data set all pages that were the result of a 404 error (indicating the the page was not found) as well as pages that were obviously no login pages (e.g., blog entries that just happened to contain the string `login` in the URL. This left us with 4,335 different benign web sites for further analysis.

To prepare the data for the following classification process, we extracted one feature vector for each page in the sets of phishing and legitimate pages. Every feature vector has one entry for each of the 18 properties that we have defined. When analyzing the feature vectors, we observed that there were many vectors that had identical values, especially in case of the phishing pages. This was the result of certain, identical phishing pages that appeared under several URLs (sometimes, an identical phishing page appeared under a few hundred different URLs). To prevent a bias in the classification model, focusing on the properties of certain phishing pages that appear frequently, we decided to include into the classification process only unique feature vectors. That is, when a number of different pages are characterized by the same feature vector, this vector is only considered once by the classification process. As a result, we ended up with 680 feature vectors for the set of phishing pages, and 4,149 for the set of legitimate pages.

Using the input data described above, we applied the J48 algorithm to extract a decision tree that can classify pages as legitimate or phishing. J48 is an implementation of the classic C4.5 decision tree algorithm [22] in Weka [31], a well-known data mining tool. We selected the C4.5 classifier for two reasons. First, we believe that a decision tree provides intuitive insight into which features are important in classifying a data set. Second, the C4.5 algorithm is known to work well for a wide range of classification problems.

Without stripping down our data to unique feature vectors, Weka created decision trees with a bias towards the properties of the most frequently appearing sites, thus delivering different results. The root node (i.e., whitelist references) stays the same. However, the count of external links gains far more importance, as it is the first node of the right subtree.

The runtimes for generating a decision tree with Weka heavily depend on the size of the input set. With our above mentioned data sets, the building of the models took 0.54 seconds for the IE data, respectively 1.29 seconds for the phishing/legitimate pages. The generation of the decision trees took about one second for the Internet Explorer tree and about ten seconds for the other one.

	Classified as legitimate	Classified as phishing
Legitimate page	4,131	18
Phishing page	115	565

Table 4. Confusion matrix for page classifier.

Running the J48 algorithm on our input data, using a ten-fold cross validation (which is set as default in Weka), the resulting decision tree has 157 nodes, 79 of which are leaves. The classification quality (confusion matrix) is shown in Table 4. It can be seen that the qualifier is quite successful in identifying phishing pages (more than 80% are correctly recognized), raising only a very small number of false alerts (18 out of 4,149 pages are incorrectly classified as phishing). This supports the hypothesis that page properties can be used to successfully distinguish between (current instances of) malicious and legitimate pages. As a result, page analysis techniques can in principle be effective in distinguishing between malicious and legitimate sites. The following Section 6.3 examines to what extent the Internet Explorer heuristics are capable of exploiting these differences to detect phishing pages.

When examining a reduced[2] version of the complete decision tree in Figure 1, it can be seen that the nodes that appear close to the root of the tree are predominantly related to a few page attributes. In particular, these are the number of whitelist references, the number of external and internal links, and the property that captures suspicious URLs. The fact that these page properties are close to the root indicates that they are most effective in discriminating between phishing and legitimate sites. Indeed, when analyzing the paths that lead to the leaves with the largest fraction of phishing pages, one can observe that the presence of many external and few internal references is evidence of a malicious site. This is even more so when these external references point to sites that are common phishing victims (that is, URLs on the whitelist). Also, a suspicious URL is a good indication of a phishing page. When looking for

[2] The tree was "compressed" by manually truncating subtrees for which there was a significant fraction of sites in only one class (either phishing or legitimate), or a small number of total sites.

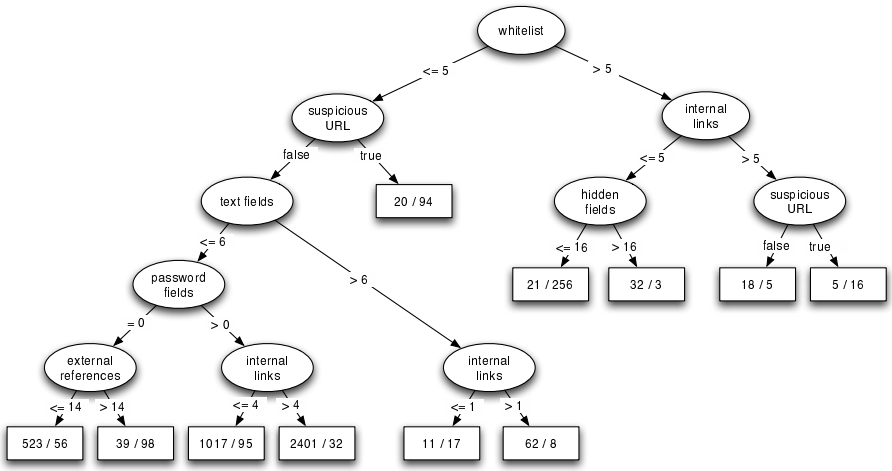

Fig. 1. Decision tree for phishing and legitimate pages.

strong evidence of legitimate pages, one will typically find many internal links and internal images (and the absence of a suspicious URL).

6.3 Analysis of Internet Explorer Heuristics

In the next step, we attempted to determine those features that the Internet Explorer page analysis heuristics considers most important to identify phishing pages. Of course, since we were only able to do black-box testing, we can only make assumptions about the inner workings of the phishing filter and how it determines the "phishiness" of a webpage.

We first used the Internet Explorer to classify our sets of legitimate and phishing pages. The process of analyzing web pages was automated, due to the large amount of data. We developed a Browser Helper Object (BHO) (i.e., IE plug-in) that was visiting pages and reporting on the results of the page analysis heuristics. After each page was visited, the BHO was used to delete all temporary files (such as browser history and cookies). This was to ensure that the next site visited would not be treated differently because of any cached information from previous pages. In addition, we had to work around the problem that the Internet Explorer offers no possibility to turn off the communication with the blacklist server. Fortunately, we could reuse the server that we previously set up to analyze the protocol between the browser and the Microsoft blacklist server. More precisely, we intercepted all blacklist requests by the browser during this analysis run and provided a response that indicated that the visited site was not blacklisted, thereby forcing IE to resort to its page analysis heuristics.

Examining the results, we observed that the Internet Explorer raised no false warnings (that is, all legitimate sites were recognized as such). This is

better than the model that we introduced in the previous section. However, the browser was also less successful in identifying phishing pages (only 1,867 of the 4,633 original phishing pages, or slightly more than 40% were correctly classified). This is likely the result of a design decision to suppress false alarms as the most important goal. In any case, even a 40% classification accuracy is valuable when considering that no false positives are raised. This is particularly true when page analysis techniques are employed as a second line of defense with a blacklisting approach. When only using page heuristics to detect phishing sites, good coverage is probably only achievable when a few false positives are tolerated (as shown in the previous sections).

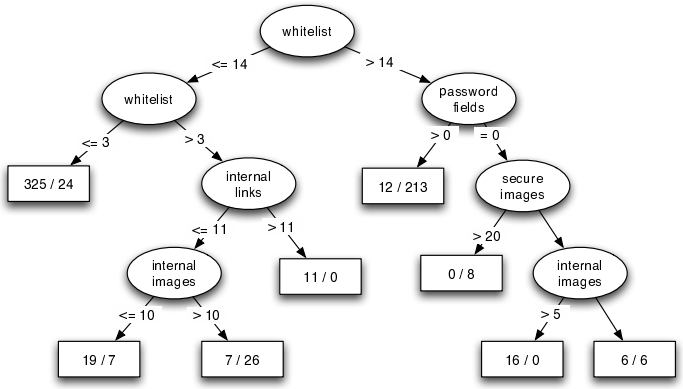

Fig. 2. Decision tree for Internet Explorer.

Once the classification effectiveness of the Internet Explorer was analyzed quantitatively, we attempted to understand which properties were most important for the decision process. To this end, we used the Weka J48 algorithm to build a decision tree based on the set of phishing pages only. More precisely, as input data set, we used the 680 unique feature vectors and the labels that IE assigns to the corresponding pages. That is, the set contained on one hand 284 feature vectors of pages that were correctly identified as malicious, and 396 feature vectors of phishing pages that were incorrectly classified as benign. The idea was to extract a model that would indicate which properties (together with the properties' values) are most important for the Internet Explorer to label a phishing site as malicious. Figure 2 shows a reduced version of the decision tree for this experiment. Note that the number of references that are in the whitelist plays a central role for classification. When the number of such references is less than four, the page is almost always classified as benign. The fact that there are nine pages with less than three whitelist references might very well be attributed to the fact that our whitelist (which is from Google) is slightly different that the one used by Microsoft. Other indicators are less significant, but one can see that the number of links to pages and images also seem to be taken into account.

7 Conclusion

In this paper, we reported our findings on analyzing the effectiveness of two popular anti-phishing solutions. We tested the anti-phishing solutions integrated into the Firefox 2 (i.e., Google blacklists) and Microsoft's Internet Explorer 7 over a period of three weeks. We fed these blacklists 10,000 phishing URLs to measure their effectiveness in mitigating phishing attacks. Furthermore, by analyzing a large number of phishing pages, we report page properties that can be used to identify phishing pages and improve existing solutions. Our findings show that blacklist-based solutions are actually quite effective in protecting users against phishing attempts and that such solutions are an important and useful component in the fight against phishing.

Acknowledgments

This work has been supported by the Austrian Science Foundation (FWF) under grants P-18764, P-18157, and P-18368 and the Secure Business Austria Competence Center.

References

[1] Firefox 2.0.0.1 Phishing Protection Bypass. `https://bugzilla.mozilla.org/show_bug.cgi?id=367538`, 2007.

[2] Anti-Phishing Working Group (APWG). APWG Homepage. `http://www.antiphishing.org/`, 2007.

[3] Neil Chou, Robert Ledesma, Yuka Teraguchi, Dan Boneh, and John Mitchell. Client-side defense against web-based identity theft. In *11th Annual Network and Distributed System Security Symposium (NDSS '04), San Diego*, 2005.

[4] David Utter. Sites Want To Hook And Gut Phishers. `http://www.securitypronews.com/insiderreports/insider/spn-49-20061114SitesWantToHookAndGutPhishers.html`, 2006.

[5] Rachna Dhamija and J. D. Tygar. The battle against phishing: Dynamic security skins. In *Proceedings of the 2005 symposium on Usable privacy and security, New York, NY*, pages 77–88. ACM Press, 2005.

[6] Rachna Dhamija, J. D. Tygar, and Marti Hearst. Why Phishing Works. In *Proceedings of the Conference on Human Factors In Computing Systems (CHI) 2006, Montreal, Canada*. ACM Press, 2006.

[7] eBay. eBay tool bar. `http://pages.ebay.com/ebaytoolbar/`, 2007.

[8] Gartner Press Release. Gartner Says Number of Phishing E-Mails Sent to U.S. Adults eearly Doubles in Just Two Years . `http://www.gartner.com/it/page.jsp?id=498245`, 2006.

[9] Google. Google Toolbar for Firefox. `http://www.google.com/tools/firefox/toolbar/FT3/intl/en/`, 2006.

[10] Google. Google Whitelist. `http://sb.google.com/safebrowsing/update?version=goog-white-domain:1:-1`, 2007.

[11] Engin Kirda and Christopher Kruegel. Protecting Users against Phishing Attacks. *The Computer Journal*, 2006.

[12] McAfee. McAfee SiteAdvisor. http://www.siteadvisor.com, 2007.
[13] Michael Sutton. A Tour of the Google Blacklist. http://portal.spidynamics.com/blogs/msutton/archive/2007/01/04/A-Tour-of-the-Google-Blacklist.aspx, 2007.
[14] Microsoft. "Microsoft Internet Explorer Privacy Statement". http://www.microsoft.com/windows/ie/ie7/privacy/ieprivacy_7.mspx, 2006.
[15] Microsoft. Phishing Filter FAQ. https://phishingfilter.microsoft.com/faq.aspx, 2007.
[16] Microsoft. Sender ID Home Page. http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx, 2007.
[17] Mozilla. Firefox 2 Phishing Protection Effectiveness Testing. http://www.mozilla.org/security/phishing-test.html, 2006.
[18] NetCraft. Netcraft anti-phishing tool bar. http://toolbar.netcraft.com, 2007.
[19] Paul Robichaux. Gone Phishing: Evaluating Anti-Phishing Tools for Windows. http://www.3sharp.com/projects/antiphishing/gone-phishing.pdf, 2006.
[20] Phishtank. Phishtank feed: validated and online. http://data.phishtank.com/data/online-valid/index.xml, 2007.
[21] Niels Provos. Phishing Protection: Server Spec: Lookup Requests. http://wiki.mozilla.org/Phishing_Protection:_Server_Spec#Lookup_Requests, 2007.
[22] Ross Quinlan. *"C4.5: Programs for Machine Learning"*. Morgan Kaufmann, 1993.
[23] Blake Ross, Collin Jackson, Nicholas Miyake, Dan Boneh, and John C. Mitchell. A Browser Plug-In Solution to the Unique Password Problem. http://crypto.stanford.edu/PwdHash/, 2005.
[24] Blake Ross, Collin Jackson, Nicholas Miyake, Dan Boneh, and John C. Mitchell. Stronger Password Authentication Using Browser Extensions. In *14th Usenix Security Symposium*, 2005.
[25] Fritz Schneider, Niels Provos, Raphael Moll, Monica Chew, and Brian Rakowski. Phishing Protection Design Documentation. http://wiki.mozilla.org/Phishing_Protection:_Design_Documentation, 2007.
[26] SpoofGuard. Client-side defense against web-based identity theft. http://crypto.stanford.edu/SpoofGuard/, 2005.
[27] Tariq Sharif. IE Blog: Phishing Filter. http://blogs.msdn.com/ie/archive/2005/09/09/463204.aspx, 2005.
[28] Verisign. Anti-Phishing Solution. http://www.verisign.com/verisign-business-solutions/anti-phishing-solutions/, 2005.
[29] W3C. IEBlog:IE7 Phishing Filter Performance Update is Now Available. http://blogs.msdn.com/ie/archive/2007/01/31/ie7-phishing-filter-performance-update-is-now-available.aspx, 2007.
[30] Liu Wenyin, Guanglin Huang, Liu Xiaoyue, Zhang Min, and Xiaotie Deng. Detection of phishing webpages based on visual similarity. In *14th International Conference on World Wide Web (WWW): Special Interest Tracks and Posters*, 2005.
[31] Ian H. Witten and Eibe Frank. *"Data Mining: Practical machine learning tools and techniques"*. Morgan Kaufmann, 2nd edition edition, 2005.
[32] Yahoo. Yahoo! AntiSpam Resource Center. http://antispam.yahoo.com/domainkeys, 2007.
[33] Yue Zhang, Serge Egelman, Lorrie Cranor, and Jason Hong. Phinding Phish: Evaluating Anti-Phishing Tools. In *Network and IT Security Conference: NDSS 2007, San Diego, California*, 2007.