DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF TORONTO

TECHNICAL REPORT #CSRG-545

iTrustPage: Pretty Good Phishing Protection

Authors:

Troy RONDA
University of Toronto

Stefan SAROIU
University of Toronto

Alec WOLMAN
Microsoft Research

December 2006

Abstract

In spite of the myriad of solutions proposed by industry and the research community to address the phishing problem, the number of phishing attacks continues to grow at a remarkable rate. This alarming trend suggests that the research community must develop new approaches to solutions that prevent phishing attacks. This paper takes a modest step in this direction.

We present iTrustPage, an anti-phishing tool that relies on user input and external repositories of information to prevent users from filling out phishing Web forms. When encountering a suspicious Web form, iTrustPage asks the user to describe the site they intend to access, as if they are entering search terms to a search engine. If the form is found in the top search results, the form is validated, and the user can proceed to fill it out. Otherwise, the user is presented with visual previews of the top search results: these are well-known sites matching the user supplied search terms. Users can either choose one of these trustworthy sites or refine their search terms.

We present a three-pronged evaluation of iTrustPage, investigating its performance, effectiveness, and ease-of-use. For this, we use previously collected traces of Web traffic, data collected from our real deployment of iTrustPage, and data collected from a controlled usability study of our tool. Our evaluation shows that iTrustPage is effective and easy to use.

1 Introduction

Phishing is a security attack based on social engineering, where the attacker attempts to obtain sensitive information, such as passwords or credit card numbers, by luring Internet users to a fraudulent Web site that emulates the appearance of the corresponding legitimate Web site. The initial contact is usually done via e-mail messages that contain a link to the fraudulent site, along with a fabricated reason for the message recipients to login. By enticing users to fill out Web forms on the fraudulent site, the attacker can record the sensitive information entered by these users, and then use this information to gain access to the legitimate site.

The damage caused by phishing attacks is enormous. Recent studies [15, 8] estimate that the cost of phishing to the U.S. economy is on the order of hundreds of millions of dollars per year, affecting more than one million Internet users in the U.S. alone. The more significant cost of phishing attacks, however, is the collateral damage: phishing erodes the trust that Internet users place in the Web as a secure e-commerce platform. Indeed, a recent study [29] found that of the surveyed people with bank accounts who do not use online banking, 40% do not trust transactions on the Web. Phishing is certainly a leading cause of this endemic distrust in the Web for financial transactions.

Although phishing is a relatively recent phenomenon, both industry and the research community have begun to investigate solutions to protect users from phishing attacks. Spam filters [5, 23, 11] have started to incorporate phishing-specific signatures to stop phishing e-mails from reaching their targets. Popular Web browsers have started to include blacklists of DNS domains and IP address ranges specific to known phishing sites [27, 24]. Several research projects [30, 18, 40, 36] have developed password management tools for the different Web sites that users access. Typically these tools must be activated by users when filling out Web forms; once activated, they provide a separate, secure form for entering a password; once entered, a secure hash function transforms these passwords before they are forwarded to Web forms. By using the DNS name of the Web site as an input to the hash function, this approach ensures that passwords sent to phishing sites are different from those sent to legitimate site. Inspired by identity theft legislation, governments are currently drafting legislation to severely punish phishers [33]. Some legislation makes it illegal for people to send phishing e-mail, even if they do not actually steal any information [2]. Some governments even force banks to compensate their customers targeted by phishing attacks [12].

In spite of all these anti-phishing efforts, the phishing threat has been gaining momentum. Phishing is growing at a phenomenal rate: the number of reported Web phishing sites grew by an order of magnitude between 2004 and 2006 [1]. Some in the banking industry claim that phishing is quickly becoming the single most important source of fraud they deal with [4]. Recent phishing e-mails are becoming very sophisticated and hard to filter; they may include personalized information, specific to the targeted user [21]. All these alarming trends indicate that our current anti-phishing efforts have done little to stop the growth of phishing. We believe, therefore, that the research community must develop new approaches to anti-phishing solutions, to reverse the current trends.

In this paper, we take a modest step in this direction. Our approach is based on two key observations:

- **Rely on user input:** We can rely on user input to help with the process of disambiguation, as long as the requests we make of the user are simple and intuitive. Certain decision making tasks are *very* difficult to automate reliably, yet are relatively easy for people to decide. For example, the task of deciding whether the visual output of two Web sites is similar can be very hard for a program to get right, yet is an easy task for a user.

- **Use external information:** We can use external information repositories on the Internet to assist the user with decision making. There are a variety of sources of information on the Internet that can be used to help establish the legitimacy of a particular Internet site and/or Web form. For example, many phishing attacks target well-known and popular Web sites that are long-lived

and have a large number of in-links from other sites. Many Internet search engines such as Google, Yahoo!, and Windows Live Search are good at identifying such sites.

In this paper, we present the design and implementation of iTrustPage, an anti-phishing solution based on these two observations. We believe that these observations can also be used in many ways beyond the specific design of our tool to address the phishing problem.

iTrustPage is a Web browser extension that implements a new approach to prevent users from entering any information into suspicious Web forms. When a user attempts to fill in a form that iTrustPage has not seen before, iTrustPage uses Google to check the PageRank [6] of that form, and if the PageRank exceeds a threshold then the form is considered valid. If not, iTrustPage intercepts the user input, and instead asks the user to enter search terms that describe the Web form they intended to visit. iTrustPage uses these search terms to perform a Google search for established Web forms, so that when the current Web form appears among the top results[1] then this form is considered valid, and the user is allowed to fill it out. If the current form does not appear in the top search results, then the user is shown visual previews of the forms that do appear in the top search results, and is asked whether any of those forms match the visual output of the current form. If a match is found, the current form is likely to be a phishing form, so iTrustPage prevents the user from filling it out and redirects the user to the established Web form.

In our initial design, the tool tried to automatically extract identifiable information from the Web forms visited by a user, such as common words or the page's main logo. iTrustPage used this information to construct Google search terms to find more established Web forms. If such a form was found, the user was transparently redirected to the more established Web form. However, we quickly realized that this initial approach was flawed: there are numerous ways for an attacker to create a Web site that appears visually similar to an established site, while still misleading our tool into extracting incorrect search terms. For example, the phisher could present a Web form as an image, an Active-X control, or a Flash script hidden in a JavaScript program, and surrounded by a large amount of bogus information. As a result, we discarded this approach.

In some cases, iTrustPage might be unable to determine whether the Web form is a phishing form: if none of the more established Web forms returned by Google are visually similar to the desired form. We believe that this case is fundamental to any phishing detection tool; all tools sometimes cannot determine whether a site is legitimate or not. Unfortunately, there is no easy way to deal with this case. One possibility is to block the user from filling out such forms. While this option might appeal to corporate system administrators, we believe that many home users would find this option annoying enough to stop using the tool. Another possibility is to raise a special warning before allowing the user to fill out the form. However, warnings have been shown to be ineffective: most users ignore them when they don't understand the implications [7, 35, 10].

In our evaluation, we will show that it is relatively rare when iTrustPage cannot determine whether a Web form is phishing or legitimate. In such cases, our tool uses an adaptive mechanism. Users are asked to revise their search terms, and the new search terms are used to find additional established Web forms. iTrustPage repeats this cycle a variable number of times, based on the PageRank of the current form. If the current form's PageRank is high, the cycle is repeated a few times; if not, the cycle is repeated many times. In the rare case that after revising the search terms, iTrustPage still cannot determine whether the form is legitimate, a warning is raised and the user is allowed to fill out the form if they want. Our preliminary deployment numbers show that this case only arises 7% of the time. Once a Web form has been deemed legitimate, iTrustPage caches this decision and the user will never be blocked from filling it out.

We present a three-pronged evaluation of iTrustPage. We start by investigating the performance of iTrustPage and we show that the overheard added by our tool is negligible. Second, we present preliminary results from our current deployment of iTrustPage. We show that iTrustPage can automatically classify as "established" 32% of the online U.S. banks, online escrow services, and online travel agencies that we tested it with. We have implemented our tool as a FireFox extension and we have made it available for download. In just 31 days, our tool has been downloaded 905 times and we have received usage logs from 388 unique IP addresses. We present an analysis of our tool's effectiveness based on reported usability statistics collected from our deployed software. Our current implementation collects these statistics and anonymizes them to protect our users identities; it also allows users to disable reporting entirely. Third, we present a controlled usability study we conducted to determine how people react to iTrustPage. We found that people can describe pages and tended to rate their tasks as "Easy".

2 Background and Related Work

In this section, we start by classifying current anti-phishing tools and techniques into four broad categories. After this categorization, we summarize the results of five previous participant studies on the behavior of Internet users when facing phishing sites. We believe the results of these studies results portray the seriousness of the "phishing threat".

2.1 Spam Filters and Blacklists

One approach relies on spam filters and blacklists to automatically prevent users from visiting a phishing site. Already there are many phishing-specific filters for popular

[1] Currently, iTrustPage uses only the top 10 results returned.

email software (e.g., Exchange Server 2003 SP2 [23], Outlook [25], and SpamAssassin [32, 11]). Many Web browsers also incorporate blacklists to prevent users from visiting specific Web sites based on the domain or the IP addresses they are served from. Microsoft's new IE7 browser, Mozilla's FireFox 2, and Opera from version 9.1 all include lists of known phishing sites. If such a site is visited, the browser will either warn the user or block the site outright [14]. Very recently, a company has started to offer a special DNS service that filters out known phishing domains [20].

Some advantages of this general approach are its simplicity, transparency, and ease of deployment. Web browsers and spam filters are already highly popular. Automatically deploying filters and blacklists is easy to setup, and much of their functionality remains transparent to most users.

While effective, this class of solutions alone will not eliminate phishing attacks. Spam filters are not perfect. Phishing sites must be quickly discovered and added to blacklists, especially since the average uptime of a phishing site was only 4.5 days in August 2006 [1]. Studies have shown that many users ignore browser warnings when they do not understand their implications [35, 10, 7]. All these reasons lead us to believe that spam filters and blacklists will have only a marginal and temporary effect on the prevalence of phishing attacks.

2.2 New Web Authentication Tools

Another approach is to invent new Web authentication schemes that replace the current approach of users entering passwords directly into forms. Different techniques have been proposed to replace current authentication protocols between users and Web sites.

One such technique is out-of-band authentication, where users are asked to login through a different channel, more secure than the Web, such as a cell-phone [28] or a virtual machine [19]. Unfortunately, some of these techniques are subject to man-in-the-middle attacks [31]. Furthermore, out-of-band techniques can be logistically difficult to deploy.

Several research projects have proposed using password managers to protect users' credentials [18, 30, 40]. Almost all these tools rely on a technique known as "password hashing" to ensure that one user never re-uses a password on more than one site. The idea is simple: these techniques use some server-specific information (for example, its SSL certificate or its domain name) combined with the user's password as input to a secure hash function, whose output is forwarded to the server. In this way, the password manager ensures that even when phished, users do not divulge their passwords; instead, they divulge their passwords hashed with information specific to the phishing site. Phishers cannot reuse these passwords on the legitimate site.

The password hashing approach is both elegant and very effective. Unfortunately, password managers have not been quickly adopted by either users or Web sites. We believe several reasons are responsible for the moderate success of these tools. First, some tools have deployment issues. For example, when resetting a password, many Web sites today send an e-mail message containing a new, perhaps temporary, password. The user must enter this new password to login, which requires disabling the password manager. The user then logs in with the temporary password, visits a form that allows users to change their password, and then re-enables the password manager to generate the permanent password. To handle this issue, many password managers use a special sequence of keys (e.g., typing the character '@' twice) or a toolbar button to activate or de-activate them [18, 30, 40]. De-activating a password manager is dangerous because the user becomes exposed to phishing attacks. Also, a recent study [7] found that many participants forget to activate (or re-activate) their password managers.

The same study [7] revealed a more subtle issue with password managers. The concept of password hashing is not easy to explain to many people, especially novice users. Some people's mental process of how online authentication works differs substantially from what password hashing does. As a result, users become frustrated, and they misunderstand when the tool is protecting them and when it is not. Ultimately, this leads to users still being exposed to phishing attacks. We believe that this is an important lesson: to be effective, a tool must be simple and intuitive, fitting most users' mental model of how Web browsing works.

2.3 New Web Interfaces

Another approach is to design Web interfaces that are less vulnerable to phishing sites. One such example is to require users to access important Web sites only through user-created labels [40]. In these cases, users have to go through an initial setup phase where they assign special labels to each of their important Web sites. From then on, as long as users visit their pages only by clicking on the appropriate label, they cannot be phished.

Another example allows users to create personalized visual clues and associate them with important Web sites. Many popular Web sites (e.g., Yahoo) have started to adopt this technique. Because a phishing site cannot know the correct visual clue, users can immediately detect when accessing a phishing Website because their personalized clue is missing or incorrect.

The main disadvantage of both these approaches is that they place the burden on the user to notice the absence of personalized clues or to never forget to access the important sites through their preset labels. These tools protect only the most diligent Web users, the ones who will always carefully check the authenticity of the Web forms they are about to fill in with their sensitive information. Unfortunately, studies [9, 35] have shown that many Internet users are not very careful when filling forms online.

A different approach is to use automatic tools to fill in forms. Such tools remove the need for Internet users to type their credentials into online forms. These tools can then perform extra security checks to make sure that the form is le-

gitimate. Web Wallet [36] is a tool that automatically fills in previously saved passwords. When the user is phished, Web Wallet detects that the current form has not been visited before. In this case, it presents the user with a list of previously filled-in forms; the user must review this list and either continue or choose one of the previous forms. If the user chooses to continue, Web Wallet issues a warning if the site has not been verified by TrustWatch [16], a site serving a list of verified Web forms.

On the surface, Web Wallet is similar to our tool: when filling out a new password field, the user is presented a list of previously filled-in forms (which are presumably legitimate). However, we believe iTrustPage is based on different insights than Web Wallet. Web Wallet relies on three mechanisms to prevent phishing: (1) automatic detection of password fields, (2) previously filled-in password fields, and (3) relying on users to notice and understand the description of a Web form (even when that description might not be available). We believe that automatic detection of password fields can be "fooled" by clever Web forms (e.g., using Active-X controls or Flash scripts). Also, we think that a good security principle is to reduce the amount of confidential information stored and managed automatically by tools. Unfortunately, Web Wallet has to maintain a list of previously filled-in passwords. Instead iTrustPage is centered around two different observations: (1) users can describe the forms they are about to fill in, and (2) these descriptions can be used to find more "established" forms on the Web; when such a form exists, most likely the current form is phishing.

2.4 Centralized Approaches

A different approach uses a centralized server that tracks when users provide the same password to different sites [13]. The main observation behind the password-tracking approach is two different sites appear to have when users with the same password on both sites, most likely one site is phishing the other. The main benefit of this approach is that servers and sites can actively deploy and use such a solution. Instead, most of the previously described approaches (and our tool as well) are deployed at the client's end rather than at the server. Many issues have been raised regarding centralized approaches, including protecting users' privacy, filtering information introduced by phishers to poison the server's data, and whether the detection of phishing sites can be done sufficiently early to rescue any potential victims.

Another approach also uses the notion of visual similarity (i.e., page layouts and style) of Web sites [22]. Site owners submit their legitimate URLs and keywords to a central server. When the user recieves an e-mail message with similar keywords, the system compares the visual similarity of the linked page in the e-mail message with the registered legitimate pages. Although this is an interesting approach, there are three key problems. First, it relies on site owners submitting their keywords and URLs, and phishers may attack this mechanism. Second, we believe it is possible for this type of heuristic to be misled by clever Web forms, which is why our tool relies on people to perform the visual comparisons. Third, the phisher may attempt to hide the relevant keywords in the phishing e-mail message.

2.5 Participant Studies on Phishing

In this section, we briefly summarize the results of five participant studies of how users behave when facing phishing sites and when using different anti-phishing tools. Overall, these studies paint a pessimistic picture of our progress against the "phishing threat": it is very easy to mislead people into divulging their credentials with common phishing attacks.

Dhamija *et al.* [9] asked 22 participants to decide whether or not 20 websites are legitimate. They found that the participants made mistakes 40% of the time. Even worse, the best phishing website fooled 90% of the participants. Warnings are ineffective: 68% of the participants "proceeded without hesitation when presented with [certificate] warnings".

Wu *et al.* [35] explored how well anti-phishing toolbars stop people from using a fraudulent Web site. They found that active warnings (e.g., pop-up warnings) are more effective than passive security cues. Even active warnings failed to prevent some of the participants from falling victim to the phishing sites. Even worse, some participants thought the warning given by the toolbar was invalid.

Downs *et al.* [10] performed a preliminary interview study including 20 participants with no computer security experience. They found that many users cannot distinguish legitimate e-mail from phishing e-mail. Many participants miss cues in the address bar, and they do not interpret pop-up messages in meaningful ways. They also found that security tools need to recommend a course of action instead of merely giving warnings.

Jagatic *et al.* [21] performed an actual phishing attack against 581 students at Indiana University in April 2005. The experiment used publicly available information to personalize their phishing e-mails. 72% of targeted students gave away their username and password when the e-mail message appeared to be from a friend. When the e-mail is sent from a fictitious person, the successful phishing rate drops to 16%. In the first 12 hours of their experiment, 70% of the total phishing responses occurred. Their findings illustrate that more sophisticated phishing attacks, such as sending personalized phishing e-mails, have very high rates of success.

Chiasson *et al.* [7] explored the usability of two password managers with 26 participants, as mentioned earlier. Their participants had difficulty building a mental model of the software – they do not have an understanding, even at a high-level, of what the software is doing. This led to frustration and misconception leading to dangerous security exposures. They also found that participants tend to dismiss warnings when their message is unclear.

2.6 Summary: Lessons Learned

In this section, we summarize the lessons learned from the above related work. The recent exponential growth in the number of new reported phishing sites [1] suggests that our current anti-phishing arsenal does not appear to be very effective. We believe that these lessons help us better identify different strategies for dealing with phishing.

- To be effective, an anti-phishing tool must be intuitive and simple-to-use. It can rely on users only to perform very simple tasks they normally perform when browsing the Web.
- Relying on users to be diligent and check for signs of suspect e-mails or Web sites can be only marginally successful.
- More sophisticated phishing attacks, such as the ones sending personalized e-mails, have high rates of success. It is difficult for spam filters to identify and eliminate personalized phishing e-mails.
- Many phishing sites are short-lived. To be effective, techniques based on detecting these sites and blocking users from visiting them must act quickly.

3 Our Approach

In this section, we present the design and implementation of iTrustPage, our tool that prevents users from filling out phishing Web forms. The design of iTrustPage is based on two observations: (1) we can rely on users to assist with the process of deciding whether a site is legitimate or fraudulent, as there are certain tasks that are very simple to ask people to do, yet are very difficult to automate reliably; and (2) we can use external information repositories, such as Internet search engine results, to assist with the process of deciding whether or not a given Web site is legitimate. User input is needed for two tasks: (1) describing search terms for a questionable Web form they are visiting to see if it is a well-known and established site; and (2) performing visual comparisons of Web forms that may be hosted by phishers with Web forms arrived at via search engine results. The external information repositories used by iTrustPage are simply the Google's search index and the PageRank information.

The biggest difference between our tool and previous anti-phishing tools is that iTrustPage relies on users to perform simple tasks to assist with validation decision making process, rather than always attempting to make the validation decision automatically. The remainder of this section presents a step-by-step description of how iTrustPage works, along with the intuition behind each step as well as presenting possible alternatives. Our goal is not to demonstrate that the algorithm or the user-interface used by iTrustPage are flawless. Instead, we hope to convince that iTrustPage's approach, which involves users performing simple tasks to assist with decision-making, is a promising alternative for preventing users from becoming victims of phishing.

3.1 Automatic Classification

The first step that iTrustPage performs is to attempt to automatically decide whether a given Web form that user visits is legitimate. First, it maintains a local cache of all previously validated Web forms visited by a user, as well as those forms manually approved by the user. In this way, iTrustPage *never* disrupts users when they revisit a Web form. For first-time visits, iTrustPage uses conservative heuristics to determine automatically that a particular form is "established": it has been long-lived, many other Web pages link to it, it is served from a very popular domain, and it has been visited by other Web users. Automatic validation makes iTrustPage easier to use: the tool remains transparent when the heuristics determine a form to be legitimate. However, using such heuristics does come with certain risks; phishers may react to iTrustPage by attempting to manipulate these heuristics to their advantage, making iTrustPage validate illegitimate sites automatically. We believe this is a fundamental trade-off between security and usability.

iTrustPage uses two very conservative heuristics to automatically check whether a Web form is "established" or not. The first heuristic is to check Google's PageRank of the Web form. Google provides a Web service that takes as input a URL and returns the URL's PageRank. The PageRank is a number between 0 (low) and 10 (high). Most sites have a PageRank of 0; only a few very popular sites have a PageRank higher than 8 (e.g., Google's search page has a PageRank of 9). iTrustPage requires a PageRank of at least 5 to automatically label a form as "established", after which it never prevents the user from filling out that form.

In addition to checking the PageRank of the actual form URL, iTrustPage also looks at the sequence of URLs accessed immediately preceding the form. If any previous URLs are served by the same site as the form, then the PageRank is calculated for each of those previous URLs, and the maximum PageRank value is used to make the decision. The intuition behind this process is that often the URL of a form at a popular site will have a very low PageRank, yet the site homepage that contains a link to that form will have a very high PageRank.

The second heuristic is to check whether TrustWatch has validated the Web form. TrustWatch is a free online service that maintains a whitelist of Web sites that they have verified to be reputable. This heuristic is also very conservative: most Web sites are not verified by TrustWatch. TrustWatch has become popular recently and it is also being incorporated into other anti-phishing tools [36].

In Section 4.2, we present several experiments showing that our heuristics are indeed conservative, allowing users to fill out only well-established Web forms, thereby confirming our hypothesis that the automatic classification component of iTrustPage strikes a reasonable balance between security and usability.

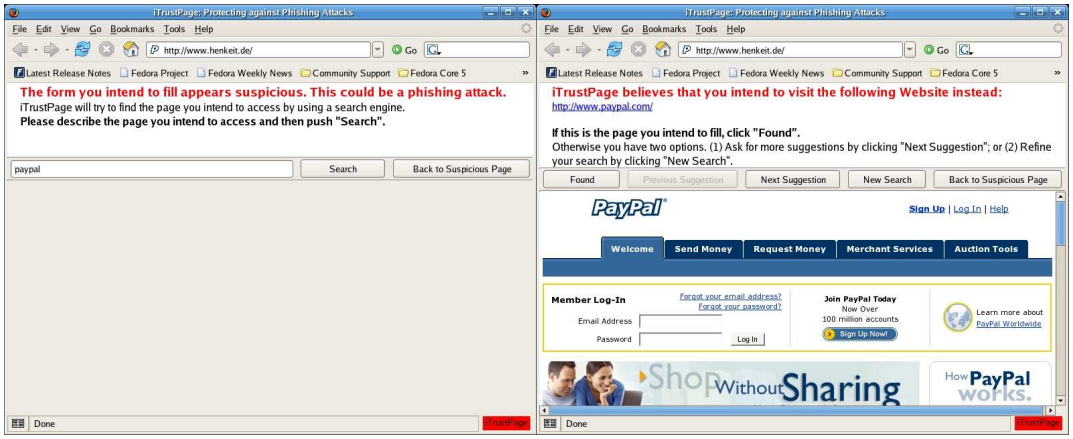

Figure 1. iTrustPage displaying its interface overlay: On the left, the user is visiting http://www.heinket.de, a well-known site phishing PayPal. Once the user enters their search terms (in this case "paypal"), iTrustPage previews the legitimate site, http://www.paypal.com.

3.2 Interactive Site Validation

When the automatic classification step fails to validate a particular Web form, iTrustPage forces the user to get involved in the validation process. Whenever users attempt to fill out a Web form using their keyboard, iTrustPage presents an overlay on the browser window that asks them to describe the form they *intend* to fill in, as if they are entering search terms to a search engine. iTrustPage uses the search terms to issue a search query using Google. If Google's search engine returns the form's Web site (i.e., its domain name) among the top 10 results, then iTrustPage infers that the Web site that serves the form is legitimate, and therefore the user is allowed to fill out the form.

The fact that the site appears in the top 10 search results means that the Google crawler indexed the site, that many other sites link to it, and that the site is most likely not short-lived. Since the user selected the search terms that led to the search results, this means that the form presumably matches the user's intent. Once a form is deemed legitimate, iTrustPage remembers that decision and will not intervene again. As future work, we could improve this mechanism by including the query results from other search engines, looking at other information repositories, such as the Whois domain registration database, and investigating whether "top 10" is the best number of search results to check (we used this number because it is the first page of results).

If the Web site that hosts the form does not appear in the top search results, iTrustPage then fetches the page contents of the top 10 search results. iTrustPage presents the user with a visual preview of those pages and asks the user if any of the search result pages are visually similar to the Web form they intended to access. If the user detects a visual similarity, then the original form is probably a phishing form. Therefore, iTrustPage immediately redirects the user away from the original form to the legitimate form found in the search results. Figure 1 illustrates iTrustPage's search interface overlay when visiting a questionable Web form (in this case `http://www.heinket.de`, a well-known phishing site spoofing PayPal). In fact, this Web form was phishing PayPal; once the user entered the search term "paypal", iTrustPage previews the legitimate site, `http://www.paypal.com`.

3.3 Revising Search Terms

In some cases, the user will not find the intended Web site among the top 10 search results. When this happens, the user is asked to refine the search and the previous step is repeated. The goal is for the user to provide a better description of the Web form they intended to access. In our experiments (described in Section 4.2), we encountered cases when users succeeded in finding the intended Web site among the top 10 search results only upon their second or third search attempt. Nevertheless, it is possible (although not very common) that users may never find the desired site among the top search results. In this case, iTrustPage is unable to determine whether or not the form is a phishing attack.

We implement an adaptive strategy in determining how many times to ask the user to refine the search before giving up and just raising a warning that states that iTrustPage was unable to determine the legitimacy of the form. Based on our experiments, we found using the form's PageRank to determine the maximum number of iterations works well in practice. If the PageRank is low, iTrustPage is more persistent, asking the user several times to refine the search; when the PageRank is higher, iTrustPage is less persistent, asking the user only a few times. Our current implementation uses the difference between our PageRank threshold for established Web sites (set to 5) and the form's PageRank; if the form's PageRank is 0, then iTrustPage asks the user to refine the search five times; if the PageRank is 4, the user is asked only once.

While we make every effort to ensure that it only occurs rarely, the problem that iTrustPage faces, of being unable to determine whether or not a particular Web form is a phishing

attack, is common to all detection tools, whether they try to detect phishing, spam e-mail, or malware. Sometimes there simply isn't enough information to reliably make the correct decision. When such cases occur, iTrustPage first raises a warning box, and then allows the user to proceed.

3.4 Implementation Details

In this section, we present additional lower-level issues related to our tool's implementation. iTrustPage is implemented as an extension to the FireFox web browser. We have tested it on several commodity OSes, including Windows, Linux, and Mac OS X. Its source code is 5,200 lines of code and it is freely available to download [3]. iTrustPage was released on October 18, 2006 and it has been downloaded 905 times already. We find these preliminary numbers encouraging.

To improve the interactive performance of iTrustPage, once a user visits a page, iTrustPage immediately prefetches the PageRank and the TrustWatch verification information, even if the user has not attempted to type anything on that page. In this way, iTrustPage can check whether the form is well-established before the user has started to fill it. Without this prefetch, iTrustPage would have to block the user from filling the form briefly while retrieving its PageRank and TrustWatch status. We implement this prefetching step asynchronously to avoid interfering with the user experience while loading the Web page.

When the user navigates to a page already deemed legitimate, we do not re-check pages in the user's navigation path until they leave the domain. For example, if the user starts at `http://www.apple.com/` and clicks on an internal link then we do not need to check the internal link's status. Once the user clicks on an external link then we re-enable checking of page statuses.

iTrustPage is configured to not block a user's interaction with a Web form until they use the keyboard or bookmark the page. iTrustPage ignores common Firefox keyboard control characters such as the spacebar, "Mac" key, control keys, and so forth. iTrustPage can optionally be configured to block user interaction on page open; to only check pages with input boxes; and to only check when the user clicks on a form element. iTrustPage does not currently deal with embedded objects (e.g., Flash and ActiveX) because it does not receive their keyboard events. We leave implementing this detail as future work.

3.5 Circumventing iTrustPage

In this section, we describe how phishers might try to circumvent the detection algorithm implemented by iTrustPage. We anticipate three types of attacks.

One way to circumvent iTrustPage is to create a phishing form hosted on a Web site with a high PageRank. There are two ways to do this. First, an attacker can break into a popular Web site and replace one of their pages with a phishing form. While this attack is possible, most popular sites are typically well monitored, and such an attack would likely not go undetected for long.

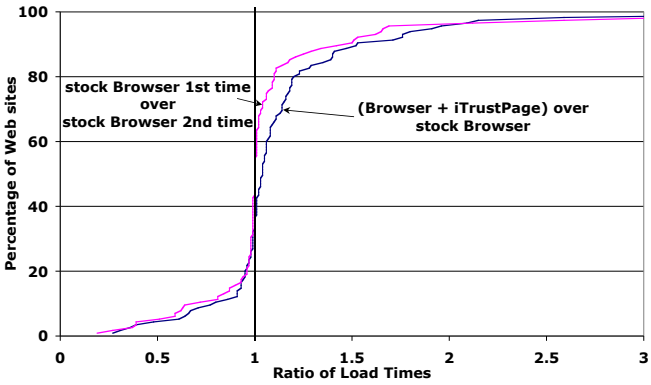

Figure 2. **Latency overhead due to iTrustPage is small:** Cumulative distribution of the ratio of latency to load a Web page with iTrustPage running to loading that same Web page in the absence of iTrustPage. To account for variability of the properties of the network, we also plot the ratio of two consecutive loads of the same page without running iTrustPage. The curves' shape is similar suggesting that iTrustPage's latency overhead is negligible.

Second, an attacker could use a "Google bomb" attack: influencing the ranking of a page by creating a large number of sites linking to it. While such attacks are not rare, this would drastically raise the costs of setting up a phishing site, whereas setting up a phishing site online today is simple and inexpensive. Also, iTrustPage could be modified to use multiple ways to determine when that a form is legitimate, such as including the results form other search engines, or using a decentralized reputation system. We plan to explore these alternatives in future work.

iTrustPage also relies on an implicit assumption: the user's browser has not been compromised. If the browser is compromised then an attacker can easily disable iTrustPage's functionality. From the beginning, we decided not to engineer against such attacks: if the browser is compromised, the user is subject to more harmful attacks, such as malware, viruses, spyware, or Trojan horses.

4 Evaluation

In this section, we take a three-pronged approach to evaluating iTrustPage. We start by characterizing iTrustPage's performance overhead on the browser when loading Web pages. Next, we measure iTrustPage's effectiveness in two ways: (1) by evaluating how often iTrustPage validates a Web site automatically, and (2) by evaluating how much effort users expend on validating their Web forms with iTrustPage. Finally, we present the results of the usability study we performed about iTrustPage in our lab.

4.1 Performance Evaluation

In this section, we characterize the performance overhead of using iTrustPage. We measure the bandwidth and latency

overhead imposed by our tool when loading a Web page. After presenting the methodology of our performance experiments, we describe our results.

4.1.1 Methodology

Because we expect that most users will run iTrustPage on their home machines, we avoided using a fast machine to measure iTrustPage's performance. Instead, we use a Pentium III 1GHz machine with 256MB of RAM, connected to the University of Toronto's network, running FireFox 1.5.0.7 in a Linux RedHat Enterprise environment. We instrumented FireFox to report when loading a page starts and when the page loading completes. We also record the amount of bandwidth consumed to fetch pages, based on tcpdump running in the background.

To evaluate the performance overhead of page loads, we used a list of 115 U.S. banks' Web sites randomly chosen from Yahoo Directory [39]. For each Web site, we record its load time and the amount of bandwidth consumed in an unmodified browser and in a browser running iTrustPage.

4.1.2 Results

For each Web site, we compute the ratio of its loading time in a browser running iTrustPage to the loading time in an unmodified browser. Ideally, in the absence of any performance overhead, this ratio would be 1. In practice however, two factors contribute to this ratio being different than 1: first, our tool adds overhead; second, the measurements are done at slightly different points in time. Therefore, unstationarity of network properties may make the loading times slightly inconsistent. To separate these two factors, we also measured the ratio between two consecutive loads of each page without running our tool.

Figure 2 plots the distribution of these ratios for all 115 pages loaded. The two curves have similar shapes suggesting that the dominating source of errors in measuring page load times is network unstationarity. The extra overhead added by our tool is negligible: on average, our tool adds 27 milliseconds to the load time of a page. As we mentioned in Section 3, iTrustPage implements its calls to the Web site's PageRank and TrustWatch index asynchronously. Therefore, iTrustPage adds negligible overhead when browsing the Web. Also, in our field trials no user mentioned that a browser appears more sluggish once it runs iTrustPage.

Although we do not present these results, we also found that the bandwidth overhead due to iTrustPage is small. On average, iTrustPage consumes 23KB extra per page, due to its queries for Google's PageRank and TrustWatch. iTrustPage's overhead was at most 40KB for 90% of the banks' Web sites.

4.2 Evaluating the Effectiveness of iTrustPage

This section's goal is to evaluate how well iTrustPage works. For this, we attempt to answer the following three questions:

1. Does the automatic component of iTrustPage correctly classify Web sites as legitimate?

2. How often can iTrustPage validate Web sites interactively?

3. How often do users need to revise their search terms?

After describing the methodology of our experiments, we present our results.

4.2.1 Methodology

We use five datasets in our experiments. Some datasets capture all Web sites visited by a group of users over time. Other datasets capture Web sites belonging to certain categories, such as online banks, escrow services, or travel agencies. In this way, we can evaluate iTrustPage on traces of typical online activity as well as on specific types of e-commerce sites. These datasets are:

Research log: A list of Web requests made by a group of 14 researchers at a research institute over three and a half months. This data is collected at a Web proxy. This data does not include any requests made over SSL. Figure 3 presents a high-level summary of this dataset.

IRCache log: A list of Web requests made by 8,714 users over six and a half months. This data is collected at several deployed Web caches. This trace also does not include any requests made over SSL. Figure 3 presents a high-level summary of this dataset.

Good sites: A list of 1,872 e-commerce Web sites. We downloaded the URLs of all Web sites posted on Yahoo Directory in three categories: online U.S. banks [39], online escrow services [38] (this category includes PayPal), and travel agencies [37]. We call this dataset "Good Sites" because it is likely that no Web site posted in these categories on Yahoo Directory is a phishing site.

Bad sites: A list of 8,489 blacklisted phishing Web sites. We obtained this dataset by combining a blacklist from Google's Safe Browsing for FireFox project [17] and a blacklist from MillerSmiles, an anti-phishing site in the U.K. [26].

Deployment log: We instrumented iTrustPage's implementation to gather usage statistics. Due to privacy reasons, we do not collect the number of Web sites visited by users, or other similar statistics not directly derived from using our tool. All information collected is anonymized before it is written to disk. iTrustPage sends this information to us once a day. Once sent, these usage logs are erased from the user's machine.

We released iTrustPage on October 18th, 2006. Over 31 days, the tool was downloaded 905 times. In total, we received 964 day-long logs from 388 unique IP addresses. Figure 3 summarizes some high-level statistics about our deployment data.

Research Log		*IRCache Log*		*Deployment Log*	
Start Date	06/20/2006, 17:07:07	***Start Date***	03/26/2006, 19:00:27	***Start Date***	10/18/2006, 13:04:16
End Date	10/20/2006, 14:51:29	***End Date***	10/15/2006, 20:00:02	***End Date***	11/17/2006, 17:35:53
# of Users	14	***# of Users***	8,714	***# of Downloads***	905
Total # of Web Pages	49,922	***Total # of Web Pages***	4,992,367	***Unique IPs***	388
Total # of Web Sites	8,439	***Total # of Web Sites***	1,591,333	***# of Daily Summary Reports***	964
Unique # of Web Sites	7,312	***Unique # of Web Sites***	460,974	***Unique Suspicious Forms iTrustPage Intercepted***	1,206

Figure 3. **High-Level summary of our datasets:** The Research log traces Internet users at a research institute. The IRCache log traces typical Internet users by capturing logs at several deployed Web caches. The Deployment Log includes the statistics captured by our tool about its use.

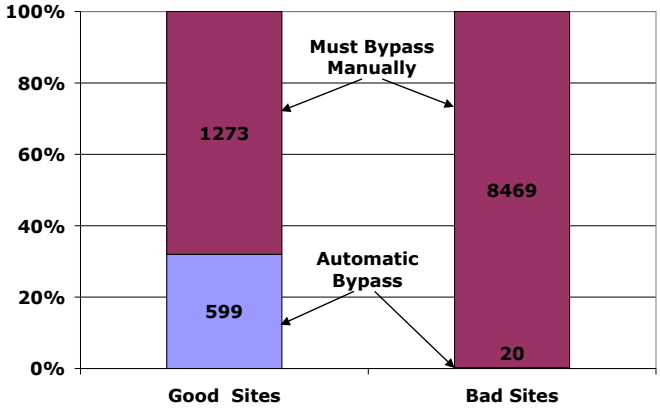

Figure 4. **Evaluation of iTrustPage's automatic classification of Web sites as legitimate:** iTrustPage can automatically validate 32% of the good sites (banks, online escrow services, and travel agencies). iTrustPage asks users to manually validate the vast majority of the bad sites (sites listed on two popular blacklists). iTrustPage also automatically validates 20 of the bad sites. Upon manual inspection, none of these 20 sites appeared to us to be a phishing site.

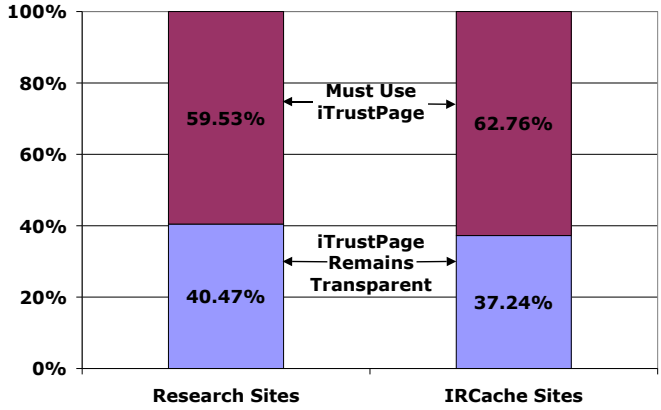

Figure 5. **iTrustPage can automatically verify many Web sites browsed by users:** Approximately 40% of all Web sites visited by typical Internet users can be automatically verified by iTrustPage.

4.2.2 Results

We start by investigating whether iTrustPage's automatic classification of Web sites as legitimate is correct. iTrustPage automatically validates any Web form whose PageRank is at least 5 or any form that TrustWatch has validated manually. We use two datasets for this experiment: the good sites and the bad sites. Ideally, iTrustPage should never validate automatically any Web sites from the Bad Sites list, otherwise its behavior is erroneous. However, iTrustPage should validate automatically some Web sites from the good sites list, otherwise this mechanism provides no benefit.

Figure 4 presents the results of our experiments. While iTrustPage's automatic classification can directly validate 32% of the good sites, iTrustPage asks for manual validation for 99.8% of the bad sites. Nevertheless, there are 20 bad sites that are automatically validated by iTrustPage. Upon manual inspection, we found that these sites appear to be legitimate; this list included `google.cn`, several Web sites serving ads, and several Web sites whose domain names listed IP addresses rather than DNS names.

While Figure 4 shows that iTrustPage can automatically validate 32% of the banking sites, escrow services, and travel agencies, it is less clear what fraction of Web forms encountered during day-to-day browsing can be automatically validated and what fraction need manual intervention. Unfortunately, due to privacy concerns, our deployment logs do not record all Web forms visited by users; instead, we only record the forms for which iTrustPage needed manual intervention. This lack of data prevents us from measuring the total number of forms visited by users. Nevertheless, we can use the Research log and the IRCache log as a first order approximation of day-to-day browsing activities. We built a simple script that simulates iTrustPage running on these logs. Figure 5 shows that about 40% of all Web sites visited by typical Internet users would end up being validated automatically. This finding suggests that iTrustPage's automatic validation is useful, leading to fewer user interruptions and better overall usability.

Once a Web form is deemed legitimate, either automat-

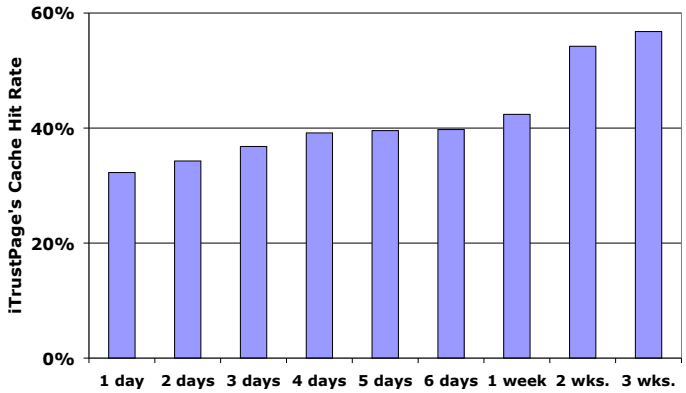

Figure 6. **iTrustPage's cache hit rate over time:** This data is collected from our deployment logs. The x-axis shows the length of the cache simulation. For each user, we record iTrustPage's cache hit rate, and the y-axis shows the median cache hit rate.

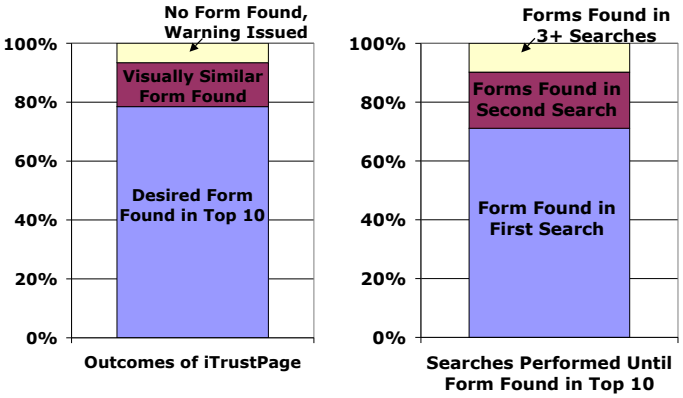

Figure 7. **Revising search terms:** On the left, we present a breakdown of the outcomes of iTrustPage's interactive site validation. On the right, we present a breakdown of the number of times users searched until the intended Web form appeared in the top 10 search results.

ically or through user intervention, iTrustPage caches this information. In this way, iTrustPage does not need to disrupt the user on any subsequent visits to the same Web form. Over time, iTrustPage builds a cache of legitimate forms on users' machines, disrupting the user less and less often. Figure 6 shows the average cache hit rates over time for our deployment logs. While iTrustPage's cache hit rate is 32% over one day, the hit rate increases to 42% after one week, and to 54% after two weeks. This confirms that iTrustPage will disrupt users less and less over time.

Finally, we examine how often users have to revise their search terms to validate Web forms, using the data collected in our Deployment logs. On the left, Figure 7 presents a breakdown of iTrustPage's outcome during manual intervention. Based on these results, we find that 78% of the time, users can find their desired form in the top 10 Google results. In this case, iTrustPage immediately returns to the desired Web form and allows the user to proceed. In 15% of the cases, the user clicked on a visually similar Web form presented by iTrustPage. These cases correspond to either a phishing form or a form with multiple URLs in different domains. Finally, iTrustPage could not validate the form 7% of the time. In these cases, iTrustPage allowed users to proceed after raising a warning.

On the right, Figure 7 illustrates how many attempts the users needed to revise their searches, for those searches that were eventually found in the top 10 search results. In 71% of the cases, the first search was successful. In 9% of the cases, users refined their search at least twice before the form was found in the top 10 results. This suggests that refining searches helps users validate their desired Web forms.

4.3 Usability Study

The goal of our usability study is to evaluate how well iTrustPage works in a controlled environment, and to observe how users interact with our tool. We start by describing the methodology of our study, and then we present our results.

4.3.1 Methodology

Before performing the usability study, we obtained approval from the Ethics Committee at the University of Toronto. The review process lasted 36 days. Once approved, we performed a study with 15 participants mainly recruited through posters and mass e-mailing. This resulted in 5 women and 10 men volunteering for the study. In terms of their occupation, we had 6 graduate students, 8 undergraduate students, and 1 non-student. Only three participants declared Computer Science as their area of study.

The participants performed the following four steps:

• After signing a consent form, the participants filled out a preliminary survey. The role of this step was to collect background information about the participants, such as whether they are students, what program they are enrolled in, how familiar they are with the Web, and whether they know what a phishing attack is.

• We presented how iTrustPage works. We described our tool's goal and we performed a step-by-step demonstration of three possible outcomes of our tool: (1) a legitimate Web site that iTrustPage can automatically validate, (2) a phishing Web site for which iTrustPage suggests alternate reputable Web sites, and (3) a legitimate, but highly unpopular, Web site for which iTrustPage cannot find any suitable alternatives, eventually raising a warning before allowing users to proceed.

• We asked the participants to perform the following six tasks:

1. Perform a common task: a simple search on Harvard's site.
2. Perform a common login: login to PayPal.
3. Perform a less common task: search for ATMs in a specific zip code on Bank of America's Web site.
4. Perform a less common login: login to the New York Times's Web site.
5. Perform a login on a phishing form: login to a Web

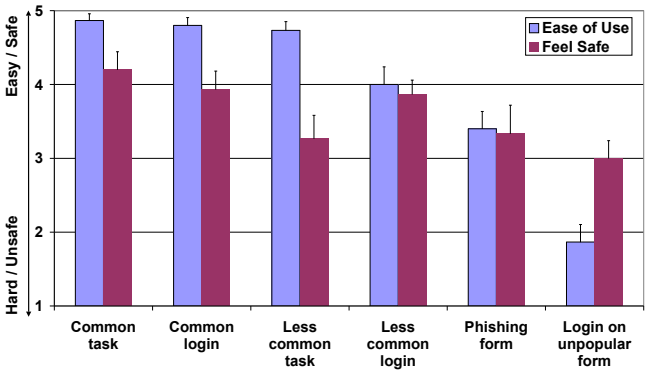

Figure 8. **Results of the questionnaire asking the participants whether iTrustPage was easy to use and whether they felt safe performing the tasks.** Overall, participants found the tool harder to use as they had to perform additional searches. However, they did not feel less safe: participants felt surprisingly consistent when performing all tasks, including the phishing one.

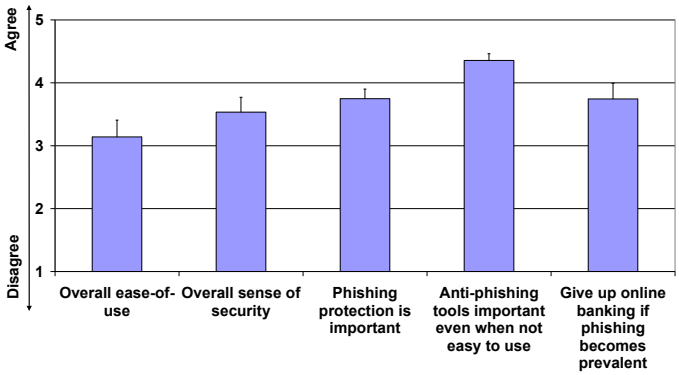

Figure 9. **Post-study questionnaire:** Participants agreed that they would trade-off usability for increased protection against phishing attacks. Also, many participants indicated they would stop banking online if phishing becomes prevalent.

form phishing PayPal. The Web form's URL displayed `http://pay-pal.com/paypal/`.

6. Perform a login on a highly unpopular form: login to our research group's WiKi page.

We gave the participants full instructions on how to perform the tasks, such as account information to login on PayPal and New York Times, and a specific zip code number (90210). We also assisted each participant individually in case they had questions. However, we did not suggest any search terms, nor did we tell them what forms to select from the top 10 Google results. However, when users did not know how to proceed, we reminded them of how our tool works.

After each step, the participants were asked to rate the difficulty of the task using a standard methodology found in usability studies: the Likert scale [34]. This is a five-point scale, asking users to whether the task was "Very Difficult", "Difficult", "Neutral", "Easy", or "Very Easy". We asked users whether they are satisfied with using iTrustPage and whether they feel safe when performing the tasks.

• Finally, we asked the participants to fill out a final questionnaire with their overall impressions about iTrustPage. Throughout the entire study, we encouraged participants to verbalize their thought process and their impressions. We recorded all their comments before, during, and after completing the tasks.

4.3.2 Results

Figure 8 shows the opinions of the participants with respect to the ease of using the tool and to their sense of security for each of the six tasks. With each additional task, the participants feel that interacting with our tool becomes more difficult. This is correlated with the difficulty of the tasks; for the early tasks iTrustPage remains mostly transparent, whereas for the final task users must keep refining their search until iTrustPage decides that it should let them proceed after raising a warning. The participants' sense of security was much less variable from task to task. This seems to suggest that participants did not become intimidated by iTrustPage's messages and instructions.

iTrustPage helped all participants detect the phishing form (task 5). While some participants already suspected the form was illegitimate (for example, some started to mention the presence of the '-' symbol appearing in the URL), others initially thought the phishing form was in fact "PayPal" (for example, one participant mentioned "Clever – I can see how easy it is to be mislead! pay-pal vs paypal.") Once iTrustPage presented a preview of the alternate Web sites, all the participants chose the correct Web site (`paypal.com`).

The final task required participants to login to our group's WiKi page. Google does not index this page. To login, participants must search for the form five times until iTrustPage allows them to proceed after raising a warning. Not surprisingly, most of the participants expressed irritation at some point during this task. One participant mentioned, "But the last task did really require some work to go through. I was not very happy to see 'You must refine your search three more times'. Over time, I would probably be annoyed and then uninstall iTrustPage." Another mentioned, " It was confusing that iTrustPage asked me to refine my search so many times because I did not know why it was requesting it. I thought that refining would search within the previous results." Many participants suggested to create a "Bypass" option instead.

We believe this task exposed the fundamental trade-off between security and usability. When we designed iTrustPage, we deliberately made the tool "irritating" if the form cannot be found in an external repository of information such as Google. We believed that by making the clearance process inconvenient, users would pay more attention to their actions. While our findings show that we succeeded in making it less convenient for users to proceed, we could not determine or quantify the amount of "extra security" gained in exchange. In the future, we might revisit this tradeoff based on the feedback that certain users would have unin-

stalled iTrustPage.

Once the participants completed their tasks, we asked them to fill in a final questionnaire. Figure 9 summarizes their answers. While the participants were neutral on how easy to use and how secure iTrustPage is, they all agreed that protection against phishing attacks is important and that using anti-phishing tools is important even when they are not very easy to use. More alarmingly, many participants indicated that they would stop doing their banking online if phishing attacks become prevalent. The participants appear to be taking the phishing threat very seriously.

At the end of the study, we had numerous informal discussions with the participants. The consensus was that they like the behavior of iTrustPage with the exception of the final task. Several of the participants mentioned that with a little bit of "polish" on the user interface, the final task could also work well. Another participant mentioned, "It's very cool... very interesting... very needed ... I don't like purchasing online... A tool that would allow me to trust [the page] would open whole new worlds."

5 Conclusions

This paper presents iTrustPage, a tool for preventing users from filling out Web phishing forms. iTrustPage relies on two key observations: (1) user input can be used to disambiguate between legitimate and phishing sites, as long as the interaction with the user is simple and intuitive; and (2) Internet repositories of information can be used to assist the user with the decision making process.

Our results show that iTrustPage is effective and easy-to-use. Specifically, we found that:

- iTrustPage's performance overhead is negligible in terms of both bandwidth and latency.
- iTrustPage automatically classifies 32% of the online U.S. banks, online escrow services, and online travel agencies as legitimate.
- 40% of sites visited by typical Internet users can be automatically classified as legitimate.
- Over time, iTrustPage's cache hit rate increases, leading to fewer user disruptions.
- iTrustPage is easy to use: when searching to validate their Web forms, users can find their desired form in the top 10 search results 78% of the time.
- Our usability study shows that users like the tool more when they experience fewer disruptions. Another finding of our usability study is that many users would stop banking online, once phishing becomes prevalent.

These findings show that iTrustPage's approach of relying on user input and on external repositories of information shows promise. In the future, we intend to examine additional strategies to incorporate these insights into other Internet security tools.

References

[1] Anti-Phishing Working Group Website. `http://www.antiphishing.org/`.

[2] Computer crimes; gathering personal information by deception (phishing), 2005. House Bill No. 2304, The State of Virginia.

[3] iTrustPage Tool, 2006. `http://www.cs.toronto.edu/~ronda/itrustpage/`.

[4] Personal Communication, 2006. Canadian Banking Sector. Toronto. We have anonymized this entry.

[5] Apache. Spam Assassin, 2006. `http://spamassassin.apache.org/`.

[6] S. Brin, L. Page, R. Motwami, and T. Winograd. The PageRank citation ranking: bringing order to the web. In *Stanford Digital Libraries Working Paper*, 1998.

[7] S. Chiasson and P. van Oorchot. A Usability Study and Critique of Two Password Managers. In *Proceedings of the USENIX Security Symposium*, August, 2006.

[8] Consumer Reports. Phishing scams grow, 2006. `http://cf.consumerreports.org/tv/index.cfm?storydate=2006-11&storynumber=1&station=w123`.

[9] R. Dhamija, J. D. Tygar, and M. Hearst. Why Phishing Works. In *Proceedings of Conference on Human Factors in Computing Systems (CHI)*, April 2006.

[10] J. S. Downs, M. B. Holbrook, and L. F. Cranor. Decision strategies and susceptibility to phishing. In *Proceedings of the Symposium on Usable Privacy and Security*, July 2006.

[11] I. Fette, N. Sadeh, and A. Tomasic. Learning to Detect Phishing E-Mails. *Technical Report CMU-ISRI-06-112, Carnegie Mellon University*, June 2006.

[12] Finextra. Korean banks forced to compensate hacking victims, 2005. `http://www.finextra.com/fullstory.asp?id=14634`.

[13] D. Florêncio and C. Herley. Password Rescue: A New Approach to Phishing Prevention. In *Proceedings of USENIX Workshop on Hot Topics in Security*, July 2006.

[14] R. Franco. Better Website Identification and Extended Validation Certificates in IE7 and Other Browsers, 2005. `http://blogs.msdn.com/ie/archive/2005/11/21/495507.aspx`.

[15] Gartner. Gartner Survey Shows Frequent Data Security Lapses and Increased Cyber Attacks Damage Consumer Trust in Online Commerce, June 2005. http://www.gartner.com/press_releases/asset_129754_11.html.

[16] GeoTrust. TrustWatch Search, 2006. http://www.trustwatch.com/.

[17] Google. Google Safe Browsing for FireFox. http://www.google.com/tools/firefox/safebrowsing/.

[18] J. Halderman, B. Waters, and E. Felten. A convenient method for securely managing passwords. In *Proceedings of the International Conference on World Wide Web*, May, 2005.

[19] C. Jackson, D. Boneh, and J. C. Mitchell. Stronger Password Authentication Using Virtual Machines. 2006. http://crypto.stanford.edu/SpyBlock/spyblock.pdf.

[20] K. Jackson. DNS Gets Anti-Phishing Hook, 2006. http://www.darkreading.com/document.asp?doc_id=99089&WT.svl=news1_1.

[21] T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer. Social Phishing. In *Communications of the ACM*, To Appear.

[22] W. Liu, X. Deng, G. Huang, and A. Fu. An Antiphishing Strategy Based on Visual Similarity Assessment. *IEEE Internet Computing, Vol. 10, No.2. 58-65*, March/April, 2005.

[23] Microsoft. Exchange Server, 2006. http://www.microsoft.com/exchange/default.mspx.

[24] Microsoft. Internet Explorer 7.0, 2006. http://www.microsoft.com/windows/ie/default.mspx.

[25] Microsoft.com. Get anti-phishing and spam filters with Outlook SP2, 2005. http://www.microsoft.com/athome/security/email/outlook_sp2_filters.mspx.

[26] MillersMiles. Phishing scames and spoof emails at MillerSmiles.co.uk. http://www.millersmiles.co.uk/.

[27] Mozilla. Firefox 2.0, 2006. http://www.mozilla.com/en-US/firefox/.

[28] B. Parno, C. Kuo, and A. Perrig. Phoolproof Phishing Prevention. In *Proceedings of Financial Cryptography and Data Security (FC)*, 2006.

[29] Research and Markets. Online banking in the united states, 2006. http://www.researchandmarkets.com/reports/344137/.

[30] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. Mitchell. Stronger Password Authentication Using Browser Extensions. In *Proceedings of the Usenix Security Symposium*, April, 2005.

[31] B. Schneier. Two-Factor Authentication: Too Little, Too Late. *Communications of the ACM. Vol. 48, No. 4.*, April, 2005.

[32] SURBL. Surbl lists, 2006. http://www.surbl.org/lists.html.

[33] U.S. Congress. Anti-phishing act of 2004, 2004. http://www.govtrack.us/data/us/bills.text/108/s/s2636.pdf.

[34] Wikipedia. Likert scale. http://en.wikipedia.org/wiki/Likert_scale.

[35] M. Wu, R. Miller, and S. Garfinkel. Do Security Toolbars Actually Prevent Phishing Attacks? In *Proceedings of Conference on Human Factors in Computing Systems (CHI)*, April 2006.

[36] M. Wu, R. Miller, and G. Little. Web Wallet: Preventing Phishing Attacks by Revealing User Intentions. In *Proceedings of the Symposium on Usable Privacy and Security*, July 2006.

[37] Yahoo. B2B / Business Opportunities / Travel Agencies, 2006. http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Business_Opportunities/Travel_Agencies/.

[38] Yahoo. Business and Economy / Shopping and Services / Financial Services / Online Escrow Services, 2006. http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Online_Escrow_Services/.

[39] Yahoo. Financial Services / Banks / U.S. States, 2006. http://dir.yahoo.com/Business_and_Economy/Shopping_and_Services/Financial_Services/Banking/Banks/By_Region/U_S__States/.

[40] K. Yee and K. Sitaker. Passpet: convenient password management and phishing protection. In *Proceedings of the Symposium on Usable Privacy and Security*, July 2006.