J Comput Virol (2007) 3:163–184
DOI 10.1007/s11416-007-0050-4

EICAR 2007 BEST ACADEMIC PAPERS

Usability evaluation of anti-phishing toolbars

Linfeng Li · Marko Helenius

Received: 12 January 2007 / Revised: 7 February 2007 / Accepted: 26 March 2007 / Published online: 16 May 2007

Abstract Phishing is considered as one of the most serious threats for the Internet and e-commerce. Phishing attacks abuse trust with the help of deceptive e-mails, fraudulent web sites and malware. In order to prevent phishing attacks some organizations have implemented Internet browser toolbars for identifying deceptive activities. However, the levels of usability and user interfaces are varying. Some of the toolbars have obvious usability problems, which can affect the performance of these toolbars ultimately. For the sake of future improvement, usability evaluation is indispensable. We will discuss usability of five typical anti-phishing toolbars: built-in phishing prevention in the Internet Explorer 7.0, Google toolbar, Netcraft Anti-phishing toolbar and SpoofGuard. In addition, we included Internet Explorer plug-in we have developed, Anti-phishing IEPlug. Our hypothesis was that usability of anti-phishing toolbars, and as a consequence also security of the toolbars, could be improved. Indeed, according to the heuristic usability evaluation, a number of usability issues were found. In this article, we will describe the anti-phishing toolbars, we will discuss anti-phishing toolbar usability evaluation approach and we will present our findings. Finally, we will propose advices for improving usability of anti-phishing toolbars, including three key components of anti-phishing client side applications (main user interface, critical warnings and the help system). For example, we found that in the main user interface it is important to keep the user informed and organize settings accordingly to a proper usability design. In addition, all the critical warnings an anti-phishing toolbar shows should be well designed. Furthermore, we found that the help system should be built to assist users to learn about phishing prevention as well as how to identify fraud attempts by themselves. One result of our research is also a classification of anti-phishing toolbar applications.

Linfeng Li is a student at the University of Tampere, Finland. Marko Helenius is Assistant Professor at the Department of Computer Sciences, University of Tampere, Finland.

L. Li · M. Helenius (✉)
Department of Computer Sciences,
University of Tampere, Kanslerinrinne 1,
33014 Tampere, Finland
e-mail: cshema@cs.uta.fi

L. Li
e-mail: linfeng.li@uta.fi

1 Introduction

Phishing has become as one of the most serious network threats [5–7]. Similar to other malicious attacks, phishing can cause loss for both financial institutions and consumers. However, unlike most crackers, phishers gain benefits by accessing credential information, instead of system or network damage. Moreover, phishing damages the trust of e-commerce.

A devastating attack does not require any emerging techniques. According to the March 2006 report of the Anti-Phishing Working Group (APWG), the most frequently used artifices are deceptive e-mails or web pages, Trojan horses and key loggers. Moreover, more than 80% of fraudulent web domains contain ambiguous names, for example, some form of target name in the URL or only IP address without host name. These ambiguous domain names are hazardous for careless consumers.

So far, there are more than 10 academic research groups and 100 of governmental or commercial organizations contributing to phishing prevention [17] in both theoretical and practical areas. On the one hand some researchers try to find the way how phishing attacks are plotted [10], and investigate

how victims decide to trust phishing scams [3]. On the other hand, other researchers intend to delve how effective anti-phishing toolbars are from a technical perspective [19]. Wu et al. have made an usability evaluation about anti-phishing toolbars [18]. Their perspective concentrates on the human behavior while using toolbars. We concentrate on the design of anti-phishing toolbars. It seems that usability evaluation of anti-phishing applications is so far rarely researched domain.

Because of the careless usability security design, phishers can easily take advantage of poor usability design [9, p. 56]. In order to offer more reliable security, anti-phishing toolbars should be easier to use. Moreover, as end-users must be able to use the toolbars and make correct choices, usability evaluation of these toolbars is important [19].

Our research objective was to find out general usability design principles for anti-phishing client side applications. Such information may result in valuable information for improving usability and security of anti-phishing applications. Based on this motivation, we conducted the heuristic usability evaluation [14] of five toolbars. However, we must advice the reader that we are not making a comparison of the toolbars in this paper. An objective comparison would require a different approach and should concentrate on assessing phishing prevention capabilities.

In this paper, we will present our evaluation and discuss the issues found during the evaluation. In the following parts of this paper, we will first introduce the features and characteristics of these five toolbars in order to make readers aware of basic functionalities from a technical perspective. After that, we will present the heuristic evaluation methodology and the evaluation results we found. Based on the results, we will give advices for improving the toolbars' usability design. In conclusion, the usability evaluation is summarized, and the impact of weak usability performance of the toolbars is discussed.

2 Introduction to anti-phishing toolbars

We found that, there exist currently four basic types of toolbars, classified by their architecture and functionalities.

1. *Toolbars based on client–server architecture and anti-phishing prevention combined with other functionalities.* These types of toolbars need to communicate with their servers, in order to protect users from being spoofed. However, these kinds of toolbars are not tailored just for phishing prevention. Instead, there are other functionalities that are not related to anti-phishing. For example, Google's Safe Browsing functionality is only one of the toolbar's features. The other features include such as Enhanced Search Box, AutoFill, etc.
2. *Toolbars based on client–server architecture and designed only for phishing prevention.* These are also based on the client–server structure, but the functionality is only phishing prevention. Therefore, users can only find phishing related functionalities from their interfaces. For example, Netcraft toolbar is designed only for phishing prevention. Even though some of its functionalities are not directly associated with anti-phishing, these are designed to support identification of fraud web pages.
3. *Toolbars installed on the local computer and detecting fraud websites by user's browsing information.* Because of the lack of the server side, these kinds of toolbars have to use the browsing information or the browsing history for detection. This kind of data cannot be managed by toolbars themselves, but by web browsers. Therefore, it is required for users to configure the browsing records carefully.
4. *Toolbars installed on the local computer and detecting fraud websites.* Different from the previous type, these toolbars must use some other methods to identify spoofing websites, like a whitelist or general detection. Compared with the third type of toolbar, users may more freely customize their own preferences, e.g. authentic web sites.

In addition to these existing toolbar types, we observed that the classification can be developed further. For example, present techniques could be combined when developing toolbars further. The classification can be based on differences in architecture, detection method and identification mechanism. Some classification variables can be:

- Is the toolbar client–server based?
- What types of lists the toolbar uses for detection (blacklist, whitelist and/or graylist)?
- Does the toolbar use local history/cache information for phishing site detection?

In this evaluation, we chose four toolbars, in addition to our own. We are aware that there are also other toolbars. However, because of limited time and resources, we picked one typical toolbar of each existing type. We selected these toolbars according to two criteria. First, the selected toolbars must be common ones and downloadable from the Internet. Second, their capabilities to detect phishing sites should be satisfactory. Based on the Zhang et al. [19], we selected Google toolbar, Netcraft toolbar and SpoofGuard, which received highest scores in the evaluation. According to the comments of the reviewers' of our paper, the Phishing Filter in IE 7 is getting used by more and more people. Therefore we included it in the heuristic usability evaluation.

We also selected our anti-phishing IEPlug to be evaluated. One might be concerned that, because we have included our

own toolbar in the evaluation, we are biased towards our own product. However, there are number of reasons for including our own toolbar. First of all, our meaning is not to place the toolbars in comparable order of goodness, but rather to find usability design principles in general level. Moreover, our aim was to improve the usability of our own application. Furthermore, anti-phishing IEPlug represents a different type of anti-phishing application. The application is not based on client–server structure and users manage their own whitelist explicitly. The warning method of the anti-Phishing IEPlug is not completely the same as with the other three toolbars. The application aims to improve user's security knowledge by actively showing the domains certificate authority (CA) information dialog. Finally, because our own application is open source application, we do not have any commercial interest.

We shall next present the toolbars included in the heuristic usability evaluation.

2.1 Google safe browsing

Google Safe Browsing (Fig. 1), is a part of the Google toolbar extension for the Firefox. It is able to alerts users based on a black list, when the web page visited is considered as a fraudulent one. There are two alternative choices for detecting fraud web pages. A user can select either "downloaded list of suspected sites" or "asking Google about each site I visit" [8]. When a user selects the first method, Firefox downloads or updates the blacklist each time, before a new Firefox window is opened. Whenever a user visits a page with Firefox, Google Safe Browsing will find out whether the page visited is in the blacklist stored locally. With the second detection method each address visited will be forwarded to a specified server maintained by Google. After the analysis, the server returns analysis results. When the page visited is considered as a deceptive one, the toolbar will stop the user's activity and give appropriate advice (e.g., stop visiting the web site, or ignore the warning). The user is also able to report mistakenly warned web pages.

2.2 Netcraft anti-phishing toolbar

The mechanism of the Netcraft anti-Phishing toolbar (Fig. 2) is similar to the Google's toolbar. It communicates with the Netcraft site's report database [13] and obtains the blacklist information. Moreover, the toolbar offers extra information concerning the page a user visits, including "RiskRating", "Since" (domains registration time), "Rank", and "Hosted server information". For example, when a user visits a page, he or she can be aware of the site's rank by following the link on the toolbar. However, this rank is based on the level of popularity, rather than security criteria. Moreover, on the toolbar, users can clearly know where the current website is hosted (in the Fig. 2, it is hosted in US). This design is considered to be helpful for users, because it is common sense

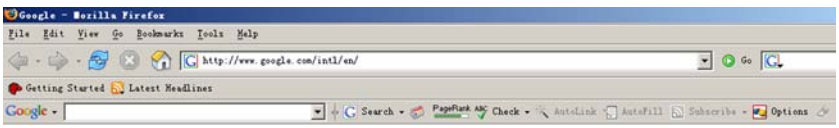

Fig. 1 Google toolbar, Safe Browsing functionality embedded

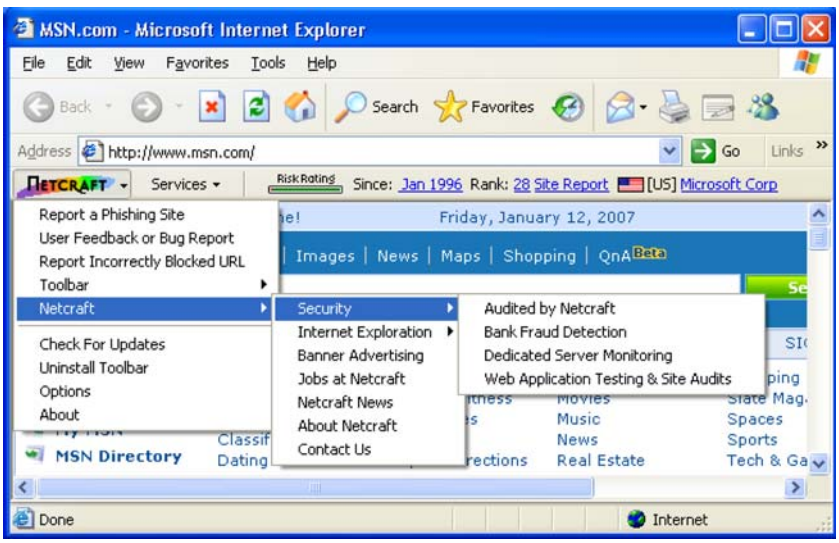

Fig. 2 Netcraft anti-phishing toolbar with a drop-down menu opened

that American Express should not be hosted, for example, in the Middle East countries.

2.3 SpoofGuard

SpoofGuard (Fig 3) is the outcome of one of the researches at Stanford University. It is compatible only with Microsoft Internet Explorer. Compared with the previous two toolbars, this one uses a different type of detection information: the Internet browsing history of the browser. There are three buttons on the toolbar: one for showing the status of the address visited, one for options of the toolbar and one for removing data collected by SpoofGuard (image hashes and password hashes).

SpoofGuard is able to warn about fraudulent web pages by checking the browsing history and other information collected, like domain name, URL, password field, image and links on the page [2]. These five kinds of information are checked in two rounds. In the first round, SpoofGuard finds the similarity between the address to be visited and the browsing history, before the page to be visited is loaded. After the page is loaded, SpoofGuard checks the password input field links on the page and images to assess the similarity of the current page and the pages the user has visited.

After these two rounds, the sum of weighed result values is computed. If the sum is greater than the "Total Alert Level" (a threshold for a warning) the toolbar will warn the user. Moreover, SpoofGuard alerts users, when they try to input the same user identity and password in different web pages. Users can change the criteria from the dialog by pressing the "Options" button. When a warning is shown, two choices are given: continue or stop visiting.

2.4 Anti-phishing IEPlug

Anti-Phishing IEPlug (Fig. 4) is a Microsoft Internet Explorer plug-in completed by the authors of this paper [12]. Likewise SpoofGuard, anti-phishing IEPlug is also a result of a University research project. The idea of this program is that a user maintains a whitelist of those domain names that she/he uses for authenticating critical operations, like e-commerce. Because the whitelist is maintained by the user or computer's system administrator, the developers do not need constant resources for updating of the program. This plug-in is able to alert about forged web pages. After users give the domain names to be detected, the plug-in begins to work. Whenever a page is loaded the plug-in at first checks whether there is a password input field on the page or not. If a password input field is detected, the plug-in will detect whether the address visited contains any domain names saved in the whitelist or not. When the address to be visited includes a keyword that is saved in the whitelist, but the actual domain is different, the plug-in will warn users. For example, a user may save "PayPal.com" to the list of domain names to be protected. When a site containing the keyword is visited, (e.g., http://www.spoofsite.com/paypal/safelogin.htm). The

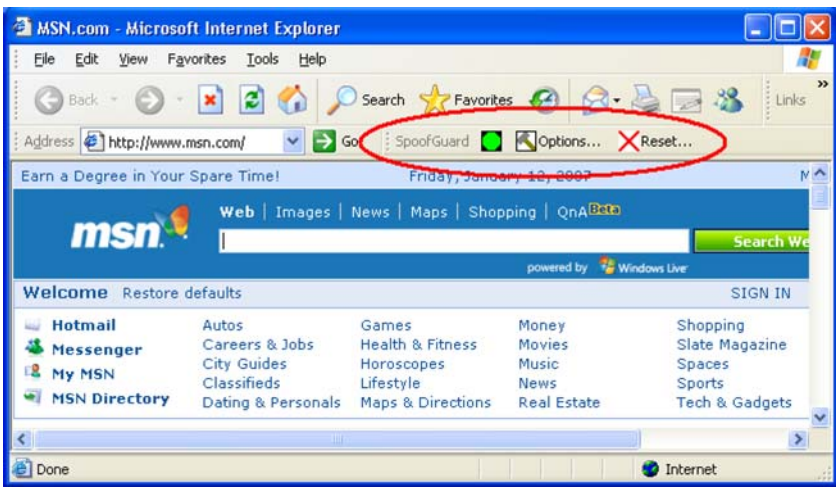

Fig. 3 SpoofGuard toolbar circled

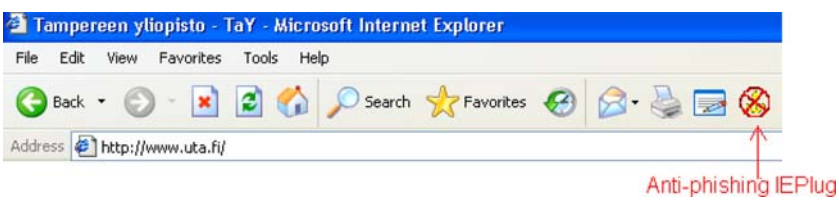

Fig. 4 Anti-phishing IEPlug button on the toolbar

IEPlug will detect this link as a possible fraud, because there is a keyword "paypal" and a user is not at the "PayPal.com" website.

In addition, the program actively shows the CA of web pages in the whitelist containing a password field There are two reasons for showing the CA information. On the one hand this shows that a user is at the authentic web page and on the other hand this method educates users.

The web pages can be warned properly based on the domain name list saved on the local machine. The plug-in offers an interface for maintaining these domain names, including adding, editing, and removing. For security reasons limited users can only add domain names to the list.

2.5 Internet Explorer 7 Phishing Filter

The "Phishing Filter" is a built-in functionality of the IE 7. The filter is based only on the client–server architecture. There are four items in the Phishing Filter's menu: "Check This Website", "Turn On/Off Automatic Website Checking", "Report This Website", and "Phishing Filter Settings". The detection mechanism can be described as the following: when a user visits a link (no matter whether it is suspicious or not), IE will firstly send the web address to a Microsoft's server. Then the server will make a query to the blacklist database and return the detection result back to the client side. If the web page is identified as a spoofing one, the browser will block the attempt to visit a web page (Fig. 5).

3 Heuristic usability evaluation of anti-phishing toolbars

For this usability evaluation, we applied Jakob Nielsen's heuristic usability evaluation method [14], which is the most common way to inspect software's usability. There are two reasons why we chose this method. First is that heuristic usability evaluation is flexible and efficient to find out potential usability issues. In addition, this method is helpful and necessary for forthcoming usability tests, because they can be based on the outcome of heuristic evaluation.

Heuristic usability evaluation specifically involves evaluators examining the interface and judging its compliance with recognized usability principles (the "heuristics") [14]. In other words, the evaluators at first define the heuristics for the user interface to be tested. If the user interface follows the heuristics, that means the interface may be preferable for users. This is a justicial method for each interface evaluated, since the principles are designed based on users' preferences before the evaluation. Moreover, in order to get reliable results, usability test specialists are required accordingly to the criteria of heuristic usability evaluation.

To guarantee the quality of evaluation results, the minimum number of evaluators is six [14]. Therefore we invited four outside evaluators, who had sufficient working experience either in usability testing or software design. In addition, both of the authors of this paper participated to the evaluation. Because both of us are familiar with phishing prevention, technical context and the design of anti-phishing applications. Furthermore, we know what functionalities

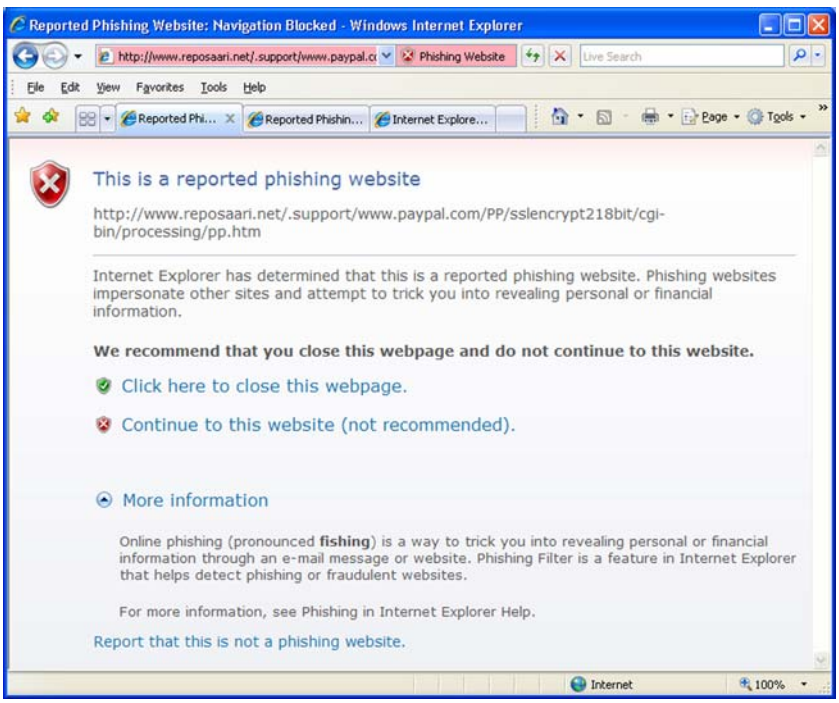

Fig. 5 A spoofing site is detected by the Internet Explorer's Phishing Filter

anti-phishing applications should contain. Moreover, we were able to find out potential vulnerabilities when using the toolbars.

The heuristics used are listed in the Appendix A [15]. These heuristics are useful for our evaluation. Firstly, this ensures that each evaluator follows the same principles during the evaluation. In this way, the evaluation can be more reliable. Furthermore, these comprehensive heuristics cover various details about the toolbar interface, which may result in more detailed evaluation results. Following the same items on the list, two evaluation results can be combined together to induce the final testing outcome. In the following parts of this section, we present the methodology, how we designed the heuristic evaluation and what were the results. Afterwards, we summarize the evaluation and what we found during the inspection.

3.1 Detailed design and implementation of heuristic evaluation

Software usability mainly focuses on some specific characteristics of software, including easy to learn (learnability), efficient to use (efficiency), easy to remember, few errors and subjectively pleasing. All of these aspects should be dedicatedly evaluated. Moreover, testing anti-phishing toolbars is not the same as testing other applications. Evaluators have to pay much attention to anti-phishing toolbars' own features during the usability inspection. According to this principle, we designed the heuristic evaluation items carefully.

Evaluation environment. We used one personal computer that was dedicated for testing usability of the anti-phishing applications. In this computer, Firefox 2.0 and Internet Explorer browsers 6.0 were installed. The operating system was Windows XP with all security updates installed. Sometimes, phishing websites contain malicious programs which may compromise the system or interfere the evaluation. Thus, it was necessary to protect the computer. For this evaluation, F-Secure anti-virus client security was installed and the system was backed up to an image file. In case the system would have been compromised it was easy to recover. Furthermore, administrators of the department were aware of the testing, network traffic was monitored, the computer was physically isolated from internal network connections and hardware firewall was present.

There were two monitors installed on the computer, one was the normal screen for the evaluators' and the other was for the observers of the evaluation. The monitors showed the same screen. The phishing websites were collected from the "PhishTank" web site [16]. Because our focus was on usability of toolbars, instead of performance, we picked up the fake sites randomly. This enabled us to see the warnings of each toolbar application in a real environment. More details of the evaluation environment are presented in Tables 1 and 2.

Table 1 Toolbar details

Toolbar	*Version number*	*Download date*
Google Safe Browsing	N/A	13 December 2006
Netcraft toolbar	1.7.0 (20061016)	13 December 2006
SpoofGuard	N/A	Dec 13th 2006
Anti-phishing IEPlug	N/A	10 December 2006
IE 7 Phishing Filter	IE 7.0.5730.11	20 February 2007

Table 2 Evaluation environment

Hardware	
CPU	Pentium III, 800 MHz
Memory	512 MB
Monitors	2 monitors, resolution 1024 × 768, 32 bit color
Keyboard	US-International English keyboard
Software	
Operating system	Windows XP, SP2
Anti-malware product	F-secure Anti-Virus Client Security 5.58
Internet Explorer	IE 6.0, SP2
Firefox	Firefox 2.0

Design of the heuristic evaluation. We collected heuristics following Jakob Nielsen's rules. In terms of these heuristics, a detailed questionnaire (see Appendix A) was implemented.

- *Visibility of the system status.* This heuristic inspects visual capability of the toolbars. Visual capability should be checked in three stages, which are visibility before checking the authenticity of a website, during checking the authenticity of a website, and visibility of the result. In each stage, anti-phishing toolbars should always keep users aware of what is going on and what is the result of identifying the web page. Moreover, response times and types should be reasonable and appropriate.
- *Match between the system and the real world.* Most of vulnerable users do not have enough knowledge of computers and the Internet. From this follows that each operation of the toolbar should be understandable and predictable for non-sophisticated users. This means that people who do not have any professional knowledge about computers and e-commerce should be able to protect themselves based on instructions or warnings from toolbars.
- *User control and freedom.* As mentioned in the previous heuristic rule, we should not expect online commerce customers having learned a lot about computers before. Toolbar designers should not assume that every user can operate each functionality of the toolbar correctly, or as

expected. Furthermore, it is necessary to provide additional functionality to undo and redo what users have done, when they recognize that there is something wrong with their operations. In addition, it should be possible for users to leave the unwanted state before the whole operation completes.

- *Consistency and standards.* This requirement comes from the system requirements. For example, it is difficult to force a Microsoft Windows user to get used to other systems, unless other systems' user interface resembles Microsoft Windows. So is the case with toolbars. The language of the toolbar should follow the platform and browser conventions as well. Moreover, advices should be consistent, when the same risk levels of suspicious web pages or e-mails are detected.
- *Help users recognize, diagnose, and recover from errors.* When users successfully pass the validation to conduct their problematic or incorrect operation, toolbars should also alert or give further advices to correct and recover from errors. This correction should be offered before, during or after users' decisions.
- *Error prevention.* Similar to the third heuristic rule, error prevention can also avoid software failures or potential problems from users' operations. Toolbars are expected to provide necessary check or confirmation before any action is committed. Different from the third heuristic rule, error prevention focuses on the validation of users' each operation and input, instead of undo functionality.
- *Recognition rather than recall.* It is required that any user, no matter who is sophisticated or not, can make a correct decision that prevents phishing without complicated operation sequences. Each warning or advice that a toolbar gives should be understandable enough. In this way users do not need to worry about being compromised due to forgetting correct instructions, even though users are misusing the toolbar.
- *Flexibility and efficiency of use.* In order to prevent phishing, typically users have to make some action, when a fraud attempt is detected. However, sometimes expert users are familiar with how to prevent specific phishing attempts, when a warning comes up and they do not want to read repeated explanations. In this case, it is necessary that flexibility and efficiency of the toolbar can facilitate experienced users' operations, and enable skipping repeated instructions, or conduct the pre-saved default operation. Obviously, flexibility may also result in faulty operation or vulnerabilities. Therefore, it is also required that users are able to return to the default settings.
- *Aesthetic and minimalist design.* This heuristic mainly concentrates on the concision of toolbars' user interface. The task of anti-phishing toolbars is to assist users to identify and stop the fraud, not e.g., commercial promotions. It is meaningful and important to make sure there is only phishing prevention related information in the toolbars. Concise and well-designed toolbar will not confuse users what should be taken into account and what should be done next, when a warning is displayed.
- *Help and documentation.* Users are not omnipotent, and they need to learn how to use different anti-phishing toolbars by themselves. In this case, user manuals, tutorials and instant help should be available with the toolbars.
- *Skills.* Phishers usually take advantage of users' shortage of network or operating system knowledge [4]. Therefore, we expect that toolbars can support, extend, supplement, or enhance users' skills and background knowledge of phishing prevention. Herein, the enhancement should be only resorted from client side, because it is not valuable to evaluate the toolbars' capability against all kinds of phishing techniques.
- *Pleasurable and respectful interaction with the user.* In this heuristic evaluation rule, we try to find out how convenient users experience usage of the phishing prevention toolbar. Both function and aesthetically pleasing value should be considered.
- *Privacy.* Toolbars are used for protecting users' confidential information from being abused or stolen. However, some toolbars also need to know personal information about users, like browsing information and contact details. This kind of information should be also carefully protected.

Besides these heuristic rules, severity of each usability problem should also be defined. Herein, we use the following rating rules provided by Tampere unit for computer–Human interaction (TAUCHI).

1. *Major usability problem*: prevents the users from using the product in a feasible manner and therefore needs to be repaired before the product is launched.
2. *Severe usability problem*: complicates the use significantly and should be repaired immediately.
3. *Minor usability problem*: complicates the use of the product and should be repaired.
4. *Cosmetic usability problem*: should be repaired for the use of the product to be as pleasant as possible.
5. *T. Technical problem*: problems marked with a 'T' are most likely due to technical problems with the product (for example, a feature that has not been implemented yet). Although they are not marked as usability problems, they will be such if left as they currently are.
6. *C. Comment*: comments (or questions) that are used for suggesting operations or point out successful implementations.

Conducting heuristic evaluation. In order to guarantee the quality of evaluation results, the whole procedure of heuristic

evaluation was carefully designed. Each evaluator inspected the five toolbars selected. Each evaluator followed the heuristic evaluation criteria presented in the prior section, in addition to the following steps:

1. *Individual preparation.* The selected toolbars were inspected once in average about 2.5 h by each evaluator. The aim was to get a general feeling about each toolbar. Before conducting the evaluation evaluators received the heuristics checklists (Appendix A).
2. *Conducting the evaluation.* The evaluation was conducted at the Virus Research Unit, University of Tampere. We observed each evaluation sequence and our main tasks were to record the findings of each evaluator. At first, Linfeng gave a basic introduction and demonstration of each toolbar in about 15 min. Then, evaluators tested each toolbar based on the preparation and checklist. At this time, they had to explain their findings of usability problems, and their findings were recorded by two authors of this paper. The time for each evaluation is given below. The Phishing Filter was evaluated separately, because reviewers of this paper asked to add the product. Therefore the evaluation time of the Phishing Filter is not included in these times.
   - First evaluator, 2 h (14.12.2006, 13:00–15:00).
   - Second evaluator, 2 h (14.12.2006, 15:00–17:00).
   - Third evaluator, 3 h (15.12.2006, 13:00–16:00). s
   - Fourth evaluator, 3.5 h (15.12.2006, 14:30–18:00).

   We also played the role of evaluators, but the evaluation method was different. When collecting data and writing, we added our ideas and findings to the final results.
   For the Phishing Filter the evaluation method was the same. The evaluation times were:
   - First evaluator, 0.5 h (20.2.2007, 11.40–12.15).
   - Second evaluator, 0.5 h (26.2.2007, 14:30–15:00).
   - Third evaluator, 0.5 h (26.2.2007, 18:00–18:30).
   - Fourth evaluator, 1 h (26.2.2007, 13:00–14:00).
3. *Gathering evaluation results.* The findings were combined into a single list and the severity of each distinct problem was rated. The following questions were discussed.
   - What were the most severe problem types?
   - What was the overall feeling about the usability of the toolbars?
4. *Reviewing the problem list.* The information from discussion is gathered and summarized for the final evaluation outcome.

4 Discussion

After the evaluation, we gathered the findings from each evaluator. Based on the evaluation and guidance of the checklist (Appendix A), we gained a number of useful usability issues. Please notice, that in heuristic usability evaluation the checklist is meant for guidance, but during the evaluation the evaluators do not need to strictly follow the checklist, but rather to bring forth each usability aspect they will find. We will next discuss the key findings from each toolbar.

4.1 Google Safe Browsing

Good usability design was observed especially when there is a phishing website detected (Fig. 6). "The dimmed area of the browser feels like something, which cannot be accessed. And the balloon can draw user's attention.", an evaluator said. There is no professional terms used, such as phishing or pharming. The advices in the warning, "Get me out of here!" and "Ignore this warning", are understandable. "Any user can easily get the points of them.", said one evaluator. The "Read more" link introduces to web forgery and phishing in technical level. This gives users freedom and is informative. The design principle in of the Google Safe Browsing seems to follow the philosophy of a good security product design. The toolbar shows a clear warning only when a phishing site is detected and otherwise the product remains silent. However, also some usability problems were found during the evaluation. First of all, there are too many functionalities on

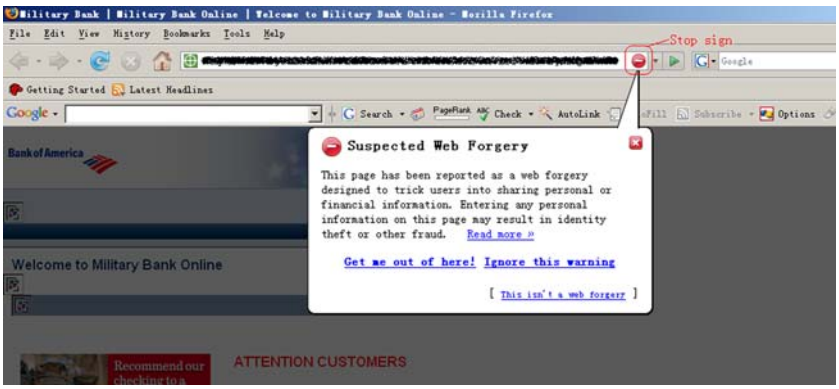

Fig. 6 A phishing site detected by the Google Safe Browsing

Fig. 7 PageRank functionality encircled

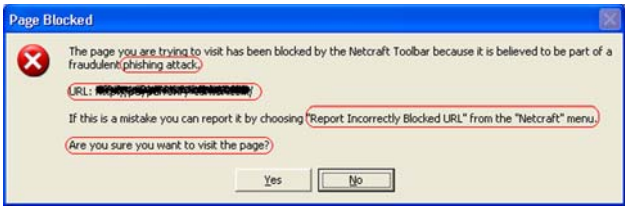

Fig. 8 Netcraft anti-phishing toolbar's warning of a phishing site

the toolbar, but no access to the Safe Browsing functionality. It is not obvious that this toolbar can prevent phishing websites. Moreover, some experienced users may misunderstand that the PageRank is part of the Safe Browsing functionality (Fig. 7). In addition, loading the phishing site regardless of the warning (dimmed area in the Fig. 6) may cause some malware stealthily being installed from the visiting phishing website. Moreover, when a user clicks the option, "Ignore the warning", there is no further warning about the danger any more. This is a problem, for example, when a user clicks the choice mistakenly. Furthermore, some evaluators believed that the phishing indicator is not consistent enough. The indicator shows up, only when a phishing website is detected. We also found that when the Google's web site cannot be loaded, (e.g., because of network traffic load) the option "Get me out of here!" will leave the user to the phishing web page, instead of being redirected to the Google's site. As a consequence a user may erroneously believe to be in a safe site. Finally, one evaluator was concerned that the warning icon (circled in Fig. 6) may be confused with the lock icon at the address bar of the Firefox browser.

4.2 Netcraft anti-phishing toolbar

Compared to the Google toolbar, Netcraft anti-phishing toolbar is designed only for phishing prevention. Therefore the toolbar's user interface is straightforward. The information about the website visited is displayed on the toolbar directly. The online tutorials are well designed.

However, the information present on the toolbar is not easily understandable. The most obvious usability problem is the implementation of the two drop-down menus "Netcraft" and "Services". Some useful options are placed unexpectedly and inconsistently. For example, "Report a phishing site" and "Report Incorrectly Blocked URL" should be services, but they are found from the "Netcraft" menu. Furthermore, structure of the menus is too complex to be easily understandable.

There are also some other minor usability issues, which may cause ambiguity. For instance, the toolbar item, "Since", is right after the "RiskRating", which may confuse users; the criteria of "Rank" is imprecise; "Site Report" does not tell whether a site is fraudulent or not. Rather the site report shows technical server information. Not every user understands the information or needs it, especially normal end-users. When a phishing website is detected, inexperienced users may not understand information in the warning dialog (Fig. 8). First of all, the popup dialog is similar to a website dialog or operating system dialog (e.g., illegal memory reference). Secondly, toolbar designers should not assume every user knows these professional terms, such as phishing and URL. Furthermore, it is not necessary to show the URL, because not every user understands the expression, especially if the web address is complex. In addition, "Report Incorrectly Blocked URL", highlighted in the warning dialog, should be clickable in order to encourage users for submitting reports, instead of forcing users to remember what they should do and where they can find the functionality. Finally, the expression "Are you sure you want to visit the page" is not clear enough. When a user quickly reads this she or he may click a wrong button. Similarly, when there is a suspicious website detected, there is no advice, except the color change of the "RiskRating" indicator.

Netcraft toolbar has a powerful website to support its services. Some of the important functionalities, such as reporting, have to resort to the website. Therefore the related web pages should be evaluated as well. As the program relies on web pages, problems will appear when there is network load or the web pages are not available. This should be taken care of in the product. For example, there could be internal help system, in addition to the web page help system. One typical usability problem is that the input field of "What's that site running" is not distinguished enough (Fig. 9). First of all, the input field is buried under other more distinguished text and advertisements. The input field can hardly draw users' attention. In addition, there is no "Submit" button. The toolbar configuration settings (Fig. 10) are not designed well from the usability perspective. First of all, the options are not grouped appropriately. For example, the appearance, and the functional settings (e.g., the level of automatic blocking) should be separated; the "Remember Details for Report URL Form" option should be more clearly grouped together with the Name and E-mail fields.

Fig. 9 Not well designed input field at the Netcraft website

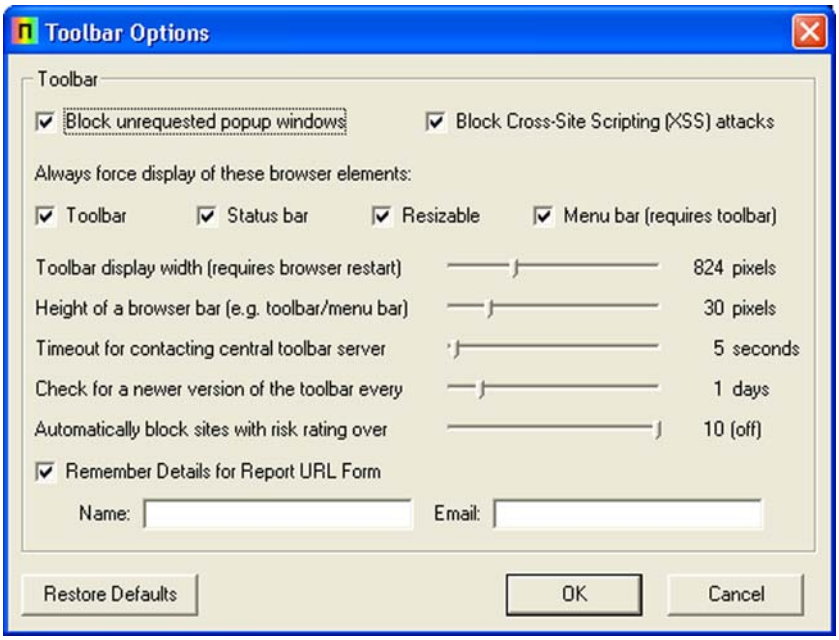

Fig. 10 Options of the Netcraft toolbar

In addition, the controls used do not follow common sense, such as checking for a newer program version.

4.3 SpoofGuard

Some advantages in the user interface were found. The traffic light is consistent enough in order to help users for identifying the risk of the current web page. Furthermore, the toolbar keeps user informed and the design is clear and concise enough.

However, from the usability point of view, there are some places to be improved. Firstly, if the web address is long enough, the Reset button and Options may be pushed outside the screen (Fig. 11).

The indicator showing identity of a web page is the color of the traffic light. However, this may be an obstacle for color-blind users. In addition, there are cultural differences as, for example, in India the color codes are different. Furthermore, the function of the reset button is unclear. Nothing seems to happen, when a user clicks the button. Users may also confuse the "Reset" button with resetting the options to the default values, but actually clicking the button will remove the image hashes and password hashes. Moreover, while the red cross

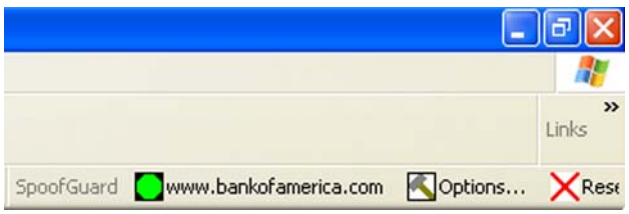

Fig. 11 Too long domain name in SpoofGuard

refers to deleting something, it is unobvious that clicking the button resets the configuration data. Similarly, the suggestion for users is not explicit enough, when a suspicious web page is detected (Fig. 12). The suggestion is likely to confuse inexperienced users, when they want to know whether they should trust the web page or not.

At last, there are some comments about the warning when a spoof web page is detected (Fig. 13). Similar to those comments about the Netcraft toolbar, some terms are too professional to be understood by normal users. Furthermore, when SpoofGuard lists suspicious places of a web page, users should be able to learn more about them, e.g., why cannot images be identical to those on another web site? Finally,

Fig. 12 Information about a suspicious web page

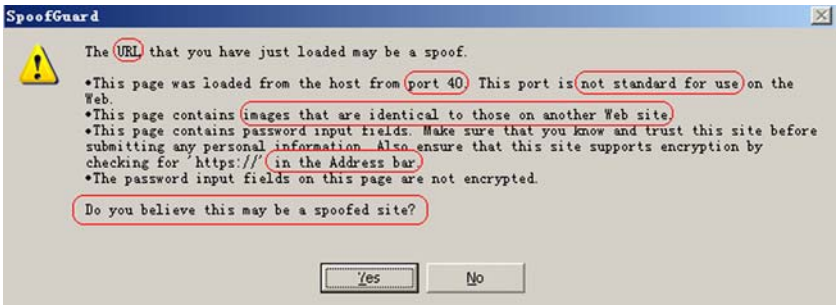

Fig. 13 Warning of a spoof web site

users can hardly understand what will happen when they click the "Yes" or "No" button. The question should be clearer.

4.4 Anti-phishing IEPlug

Likewise SpoofGuard, Anti-phishing IEPlug is also a result of a University research project. Compared to the previous three toolbars, the user interface of this application on Internet Explorer toolbar is very simple, only one button. Furthermore, the idea to maintain only a whitelist was considered convenient, because it gives power to users and does not require constant updating. However, some parts of the program needed to be improved. The popup messages were used too frequently. Sometimes, they were annoying. For example, when Anti-phishing IEPlug is installed successfully, a message was shown each time Internet Explorer is opened (Fig. 14). The reason for showing the dialog was to show a user that the program is active and protecting the user. A better solution would be to show the program status e.g., as an icon on the toolbar. Another example is that when Anti-phishing IEPlug adds a domain name, there is no need to remind users with a popup message. Furthermore, the message text is too technical. End-users are likely to be confused.

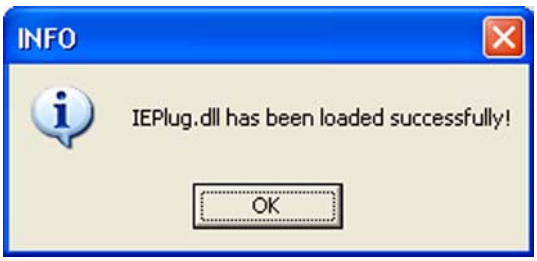

Fig. 14 Popup message of the Anti-phishing IEPlug

The interface of the domain name configuration (whitelist) dialog was not satisfactory either (Fig. 15). First of all, the title of the dialog does not make sense. A title should represent the purpose of the dialog. Likewise with other toolbars, some terms were too difficult to understand. The edit dialog for domain names was not long enough and adding domain names was not consistent. The address was shown in the edit box, but only domain name was added to the whitelist. Furthermore, the dialog should be stretchable so that users can view as many domain names as possible at one glance. The buttons were not well designed. There were two "Close"

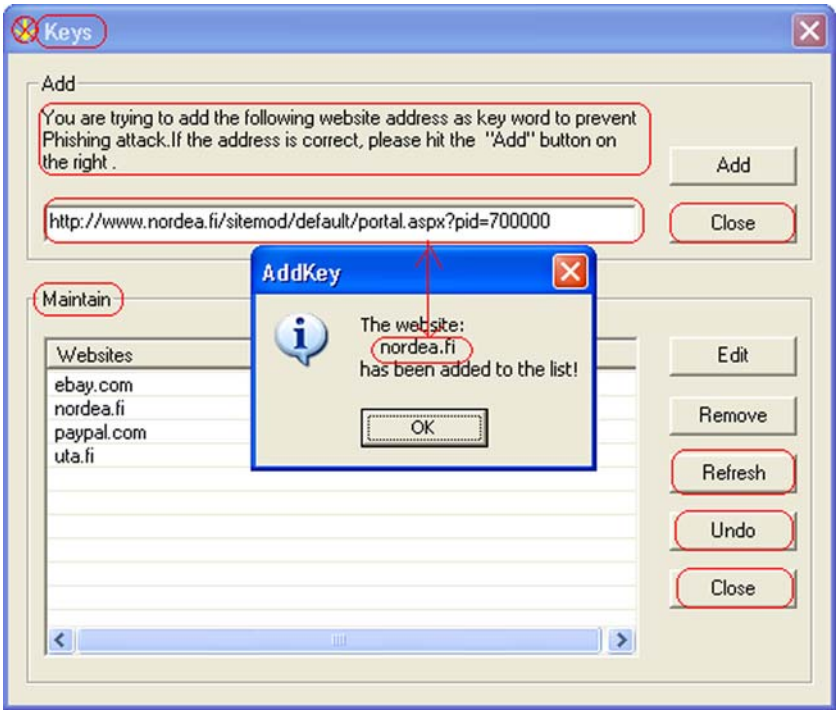

Fig. 15 Adding a domain name to the whitelist

Fig. 16 Warning of the Anti-phishing IEPlug

buttons and there was no feedback, when a user clicks "Refresh" or "Undo".

The warnings of the Anti-phishing IEPlug were also problematic from the usability point of view (Fig. 16). There were too many technical terms and there was no further information available to users. There should be a well designed help system to give users more information. In addition, it would be convenient that popular authentic domain names were presaved to the whitelist. This feature could be implemented with client–server architecture, by collecting websites from user's whitelists.

4.5 Internet Explorer 7 Phishing Filter

The Phishing Filter is embedded functionality of the IE 7. This kind of design should be able to co-operate with other components of the IE 7. The pros of Phishing Filter include colored address bar (a constant warning indicator), straight and informative warning, off-line help documentation, as well as a well designed interface to report suspicious and falsely detected web sites. However, there are also some places to be improved. In the following, we will present the usability problems found during evaluation.

The warning can successfully stop users from visiting identified phishing web pages (Fig. 5), but not everything is satisfactory. At first, the icons for two options on warning page are not evident enough. These icons are the same as those in the dialog "Turn On/Off Automatic Website Checking". However, the functionalities are not related. Moreover, it would be better to give the criteria for phishing site detection.

A serious usability issue is that when a user clicks the choice "Click here to close this webpage" (Fig. 5), there is a confirmation that prevent users from closing the web page directly. This design may mislead users to think their action is risky. However, there is no extra warning when a user clicks the choice to visit the phishing web site detected.

"Check The Website" is a sub-functionality of the Phishing Filter, which can report the authenticity of the current web page. However, the information given is not good enough. First of all, when the current page is not in the black list, it cannot tell users how to check the authenticity manually. Furthermore, the instruction of how to "Report This Website" is not easy to remember. Instead, a link should be given to make users report the current page.

One serious security related problem we found is that the dialog (Fig. 17) is still shown, even when the network connection is disabled. This will mislead users and what is worse,

Fig. 17 The dialog when a user clicks "Check The Website" and it is not a reported phishing website

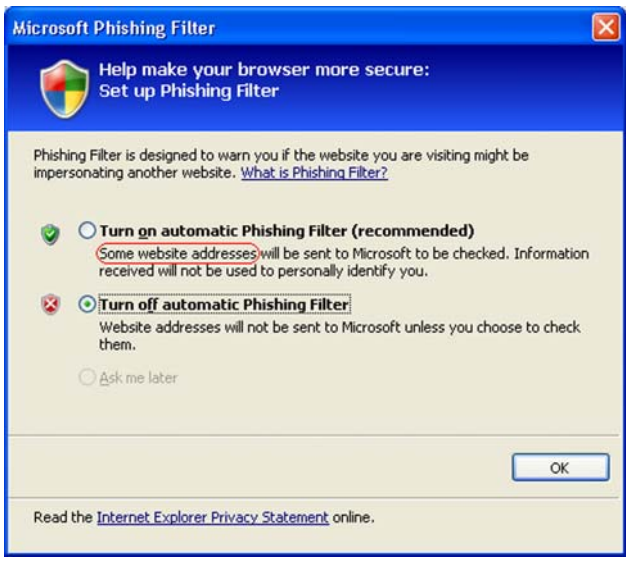

Fig. 18 The dialog to "Turn off the Automatic Website Checking" while the current setting is ON

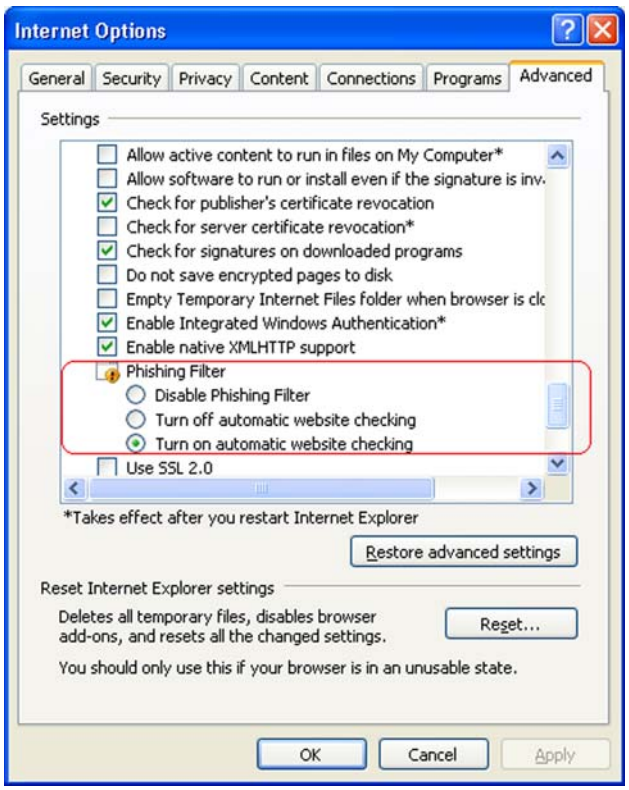

Fig. 19 The dialog to change the settings of the Phishing Filter. The settings of the Phishing Filter are circled

if the network connection fails, it seems that the Phishing Filter will give false advice to a user. "Turn On/Off Automatic Website Checking" is problematic from the usability point of view. First of all users are likely to confuse this functionality with the "Automatic Website Checking" setting. Secondly, when a user wants to turn off the automatic checking (Fig. 18), the status seems to be already at "Turn off…". This design in meant to facilitate the operation, but users are likely to be confused.

Moreover, the instruction on this dialog is also ambiguous. For example, it is said "Some website addresses…" (circled in Fig. 18). A user may wonder why not all addresses will be sent. This may make users feel insecure. A better solution would be to show these choices with the "Phishing Filter options" along with all the other possible choices. Furthermore, there are more than these three settings for the Phishing Filter. All settings should be present in one dialog. The menu item "Report This Website" allows a user to submit a suspected phishing web site to the server at Microsoft. There are three steps, in order to successfully submit. Even though users may understand these steps, there are some minor problems. For example, there is no way to recover a mistaken submission. Moreover, there is too little information about how this submission works or helps other users. There is a challenge shown to the user, but the letters were sometimes difficult to interpret correctly.

"Phishing Filter Settings" allows a user to switch on/off automatic checking and to disable the filter. However, when a user clicks this functionality, a general "Internet Options" dialog is shown (Fig. 19), instead of the Phishing Filter's setting dialog. Furthermore, there is a long list on this dialog,

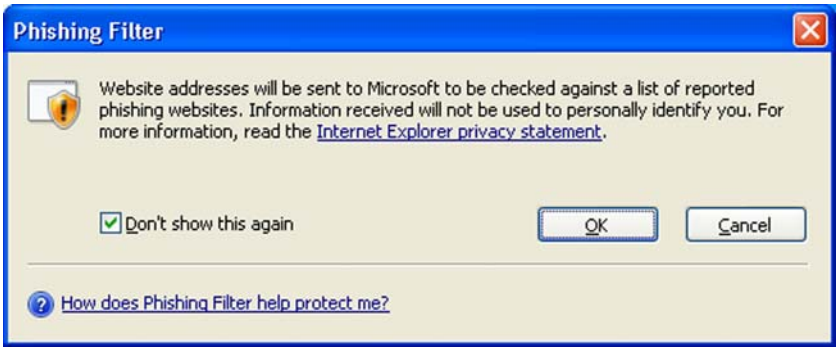

Fig. 20 The dialog shown when a user clicks "Check This Website"

and the settings for the Phishing Filter are listed nearly at the end of this list. This design really disturbs users. It would be better to put these settings to a separate options dialog. The privacy issue is also taken good consideration by Microsoft. When a user uses "Check The Website" of Phishing Filter for the first time, the "Internet Explorer privacy statement" is shown explicitly (Fig. 20). However, it would be possible to store the black list locally in a similar way as the "Google Safe Browsing" functionality allows. In this way users could have more trust on privacy protection, because there would be no need to send browsing information to an external server. There are also some general usability problems. The Phishing Filter may not be easy to find from the "Tools" menu. Furthermore, there is no direct entrance or button to the "Check This Website" functionality. It is not even possible to add a button to a toolbar. In addition, the help documentation should be accessible from the sub-menu of the "Phishing Filter".

Table 3 Statistics for Google Safe Browsing

	Major	Severe	Minor	Cosmetic
Visibility	2	1	0	3
Matching the real world	1	1	1	0
User control & freedom	1	3	1	0
Consistency & standards	0	1	1	0
Help user recognize	1	0	0	1
Error prevention	0	1	0	0
Recognition	2	0	0	0
Flexibility	1	3	1	0
Aesthetic design	0	0	0	0
Help & documentation	1	1	2	1
Skills	1	2	1	0
Pleasurable interaction	0	1	0	0
Privacy	0	0	0	0

4.6 Statistics

After the evaluation, we firstly reviewed the comments and then completed the final heuristic checklist. Based on the heuristic usability criteria, we assigned the severity level to each usability problem. After that, we collected the usability problems of each anti-phishing application and constructed the following statistics tables. According to the Nielsen's principles, six evaluators are sufficient to find most usability problems [14]. Therefore, the sample size is large enough for the heuristic usability evaluation.

We cannot give detailed heuristic evaluation results, because of the length limitation. Therefore we present general statistics of the usability problems found and their severity levels. The statistics are not meant for comparing the toolbars' usability performance. Instead we want to show that there exists a number of usability problems in the toolbars evaluated.

We would like to advice the reader that there are also limitations with these statistics. First of all, these statistics are very rough evaluation results, which cannot reflect every usability problem precisely. Even though the heuristic checklist was designed beforehand, the entire evaluation is based on the evaluators' personal opinion. Therefore the result of the evaluation may not be comprehensive enough. For further research, it is necessary to conduct a usability testing in order to collect users' experiences and feedbacks directly. The statistics for their evaluation results are listed below (Tables 3, 4, 5, 6, 7):

4.7 Suggestions for improving usability of toolbars

According to the comments and statistics constructed from the evaluation, we made a number of findings for anti-phishing client side application usability design. Generally, we found that there are three basic components that should be well designed: the main user interface of the toolbar, warnings, and help system. We will next discuss our key findings in these components.

Table 4 Statistics for Netcraft anti-phishing toolbar

	Major	Severe	Minor	Cosmetic
Visibility	3	5	0	0
Matching the real world	5	5	0	0
User control & freedom	1	2	1	0
Consistency & standards	3	4	0	1
Help user recognize	3	0	0	0
Error prevention	2	1	0	0
Recognition	3	0	0	0
Flexibility	0	5	1	0
Aesthetic design	1	2	0	0
Help & documentation	0	1	1	0
Skills	2	1	1	0
Pleasurable interaction	0	3	0	0
Privacy	0	1	0	0

Table 5 Statistics for SpoofGuard

	Major	Severe	Minor	Cosmetic
Visibility	1	1	0	1
Matching the real world	4	2	0	0
User control & freedom	1	2	0	0
Consistency & standards	2	3	0	0
Help user recognize	1	1	0	0
Error prevention	2	0	0	0
Recognition	0	2	0	0
Flexibility	0	5	1	0
Aesthetic design	1	0	0	0
Help & documentation	5	5	0	0
Skills	2	1	2	0
Pleasurable interaction	1	2	0	0
Privacy	2	0	0	0

Table 6 Statistics for Anti-phishing IEPlug

	Major	Severe	Minor	Cosmetic
Visibility	1	5	3	0
Matching the real world	1	4	0	0
User control & freedom	1	2	0	0
Consistency & standards	3	2	0	0
Help user recognize	1	0	0	0
Error prevention	1	1	0	0
Recognition	2	3	0	0
Flexibility	0	4	2	0
Aesthetic design	1	2	0	0
Help & documentation	3	6	1	0
Skills	3	2	0	0
Pleasurable interaction	1	3	0	0
Privacy	0	0	0	0

Table 7 Statistics for IE7 Phishing Filter

	Major	Severe	Minor	Cosmetic
Visibility	3	2	1	0
Matching the real world	1	2	0	0
User Control & freedom	2	2	1	0
Consistency & standards	1	0	0	1
Help user recognize	2	0	0	0
Error prevention	1	0	0	0
Recognition	3	1	0	0
Flexibility	3	2	0	1
Aesthetic design	1	0	1	0
Help & documentation	1	2	1	0
Skills	2	3	0	0
Pleasurable interaction	1	2	0	0
Privacy	0	0	0	0

1. *Main user interface of the toolbar.* According to our perceptions, the main user interface of the toolbar is very important. First of all, the status of the toolbar should be shown appropriately. This means that whenever browsing a web page, the user should be able to easily observe what toolbar is doing and whether the current web page is authentic or not. Secondly, the anti-phishing client side application interface should be simple enough so that it is easy to understand and it does not take too much space from the browser's interface. Of course, frequently used and important functionalities, such as configuration settings and viewing the website identity analysis result, reporting a suspicious or misjudged web page, should be convenient enough to be found. In this regard, some parts of the SpoofGuard's user interface design could be a very good example, such as the traffic light indicator and the Options button. These buttons are informative and make functionalities easily accessible.
2. *Warnings.* Considering the lack of reliable strategy to detect the fraud, the warnings of the application need to be carefully designed. It is important that a user is able to react correctly when a fraud or suspicious web page is found. According to the evaluation by Zhang et al. [19] observation, the false and undetermined detection is not a minor issue. It would be problematic if a user relies only on these toolbars with fixed detection algorithms. Therefore, there should be at least three levels of security indication: the warning for detected web forgery, the warning for a determined suspicious page and the indication for innocent or authentic page.
   The warning of the Google Safe Browsing is a good example for showing the web forgery. The Google's warning can stop users' faulty visits properly. The warning for

suspicious page can be the same as the one for the forgery. The differences between them could be on the given advices and their indications. For example, there may be only one advice (stop visiting) available for the forgery, and the indicator for the warning could be a stop sign. Furthermore, there could be two further advices (stop visiting, or check authenticity manually), when a suspicious page is found. The indicator should not be as strong as the one for detected page,(e.g., an exclamatory mark). The undetermined page requires to be notified to users as well. When this kind of page is found, the instant help documentation or instructions are needed in order to help users identify suspicious pages manually. Additionally, in order to be consistent innocent pages should be indicated, respectively. For instance, the indicator could be shown at the same location of other levels of phishing warning indicators. Finally, a double warning should be used in case an erroneous choice is made. If a user accidentally selects a choice that leads to visiting a phishing website, the second warning should be available to correct the mistake.

3. *Help system.* Compared to other software, the client side anti-phishing application must be able to help users at any critical occasion. These occasions include when users may select a dangerous choice, when they are confused by some terms, and when they want to learn how to identify a correct service manually. Regarding the efficiency and convenience of help, the ways of showing help for different occasions may not be the same. For example, when a user tries to find further advice, instant help system is needed. Another example is when a user should find out consequences of different choices when a warning is present. However, the text, which may help the user to understand some terms or consequences of choices, cannot be put together with the warning. It must be remembered that too much information will confuse users. For the other two occasions, the online help documentation would be better, because there can be much more information. The help system of the Netcraft toolbar could be a valuable example.

Finally we have two general results. First of all, it is beneficial to apply whitelist and blacklist methods together. Even though blacklist based application is able to correctly detect the identified (or reported) phishing sites, it is still possible to fail to warn non-identified or non-reported ones. Relatively, whitelist contains information of the websites to be protected, but all phishing websites cannot be identified. Therefore, more protection could be gained by combining these two kinds of lists into an anti-phishing application. Actually this is currently possible by using a blacklist based application together with a whitelist based application. For example, the Google Sage Browsing, Netcraft or Phishing Filter could be used together with the Anti-phishing IEPlug.

The other general finding is that, anti-phishing client side applications should not rely merely on the Internet, because sometimes the online traffic is not good enough. For example, when a user chooses an option to leave a phishing website, a toolbar could direct the user to a safe page. However, if the connection fails at that time (we met this occasion during the evaluation), the user may stay on the fraud website. This places the user at unnecessary risk. Therefore, anti-phishing applications had better redirect to the locally saved page or in some other way handle the possible fault with the Internet connection. Furthermore, it should be taken care of that the online help systems and reporting systems, which rely on the Internet connection, may not work all the time.

5 Conclusions

In this paper, we presented a design of heuristic evaluation of five typical anti-phishing applications, and discussed our findings from the evaluation. As far as we know, this evaluation is novel usability research in the phishing prevention domain. We found some important usability issues, which could be helpful for further improvement of anti-phishing toolbars. Furthermore, the heuristics checklist could be reusable for future testing as well.

However, there are also some limitations in the evaluation. Due to the natural drawbacks of heuristic evaluation, we cannot get precise and direct users' feelings on using these toolbars. Moreover, because of the limited number of evaluators, not every usability problem was found. With conducting the future usability test, those drawbacks could be overcome with larger resources. In addition, we failed to indicate whether showing actively in our application the CA is usable or not. Finally, the number of evaluated application types is limited. When we were conducting the evaluation, we realized that there could be more than four types of anti-phishing client side applications.

Despite of the limitations of this evaluation, there are some contributions to the anti-phishing research domain. We succeeded to construct a heuristic checklist, and found out some key usability issues based on the evaluation, including how to implement an appropriate warning, how to warn a user in an understandable and polite manner, and what are essential components of client-side anti-phishing applications. All of these may facilitate the future usability design and be a basic guidance in the anti-phishing usability domain.

Acknowledgements We would like to thank the evaluators Lauri Immonen, Lily Wen Lin-Marsalo, Wenfeng Liu and Markku Myllylahti for their participation to the evaluation and for their effort and valuable input for our paper. Without their help our paper would not have been possible. We would also like to thank the EICAR program committee

and especially Vlasti Broucek for valuable and constructive comments to our paper.

A Appendix A: Toolbars heuristic evaluation: checklist

A.1 Visibility of toolbar status

The toolbar should always keep users informed about what is going on, through appropriate feedback within reasonable time (Table 8).

A.2 Match between toolbar and the real world

The toolbar should speak the user's language, with words, phrases and concepts familiar to the user, rather than software oriented terms. Follow real-world conventions, making information appear in a natural and logical order (Table 9).

A.3 User control and freedom

Users should be free to select and sequence tasks (when appropriate), rather than having the toolbar do this for them. Users often choose toolbar functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Users should make their own decisions (with clear information) regarding the costs of exiting current work. The toolbar should support undo and redo, when user choose advices given by toolbars (Table 10).

A.4 Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions (Table 11).

A.5 Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (NO CODES; Table 12).

A.6 Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place Table 13).

Table 8 Visibility of toolbar status checklist

#	Review checklist	Yes No N/A	Comments
1.1	Does every display begin with a title or header that identify itself?		
1.2	Is the toolbar status shown before visiting any new web page?		
1.3	Is it easy to find which operations are available before visiting any web page?		
1.4	Is the toolbar status shown during verifying legitimacy of web pages?		
1.5	Is there any suggestion on what user should do during waiting for verification result?		
1.6	If there are observable delays (greater than fifteen seconds) in the toolbar's verification response time, is the user kept informed of the toolbar's progress?		
1.7	Is the web page legitimacy analysis result shown properly after showing the content of web page?		
1.8	If error occurs because of users' mis-operation, is user able to see the field in error?		
1.9	Is there some distinguished form of toolbar feedback for warning, when the fraud is detected?		
1.10	Is it clear to know which items in the dialog or toolbar are selectable?		
1.11	Are every two items separated properly on the toolbar?		
1.12	Are the buttons on the toolbar separated obviously?		
1.13	Is there any clear explanation from toolbar before significant operation (e.g., decide to keep visiting the warned suspicious web page)?		
1.14	Are the fraud detection response time less than 1 s, regardless connection delay?		
1.15	Is the used terminology consistent with anti-phishing domain?		
1.16	Can image on button express the function of it correctly?		
1.17	Can user distinguish different GUI controls from each other (e.g., Drop-down list does not look like a button) ?		
1.18	Can users know what operations are available, when the fraud is found?		
1.19	Can users know whether the visiting web page is deceptive one or not, after toolbar's verification?		

Table 9 Match between toolbar and the real world checklist

#	Review checklist	Yes No N/A	Comments
2.1	Are icons meaningful and concrete?		
2.2	Are the menu items and buttons on the toolbar ordered in the logic way, giving users what will be done after selection?		
2.3	If there is a visual cue (e.g., images on buttons), does it follow real-world conventions?		
2.4	Are there any obvious differences between selected and unselected?		
2.5	On user input field, are tasks described in terminology familiar to users?		
2.6	Are the questions understandable, which are given by popups of toolbar?		
2.7	Do menu choices or words on buttons have readily understood meanings?		
2.8	Are menu items parallel grammatically?		
2.9	Do commands follow the language in daily life, instead of computer science domain?		
2.10	If in need of input, is it available to give uncommon letters?		
2.11	Is key function of toolbar labeled clearly and distinctively?		
2.12	Are fields on the window separated appropriately?		
2.13	Are fields on the window grouped appropriately?		
2.14	Are buttons or menu items grouped appropriately?		

Table 10 User control and freedom checklist

#	Review checklist	Yes No N/A	Comments
3.1	Can users check legitimacy of web page at any time when they want?		
3.2	Can users conduct several same functional operations together, instead of repeating them one by one?		
3.3	Is there "Undo" function provided, or can user cancel the former operation which can cause serious consequence?		
3.4	Are menus on the toolbar broad (many items on a menu) rather than deep (many menu levels)?		
3.5	When manipulating sensitive data (e.g., delete or add the fraud web page name from black list), can any user have the access?		
3.6	Can users set their own preferred layout of toolbar?		
3.7	Can users set their own preferred warning methods?		

A.7 Recognition rather than recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of toolbar should be visible or easily retrievable whenever appropriate (Table 14).

A.8 Flexibility and Minimalist Design

Accelerators-unseen by the novice user-may often speed up the interaction for the expert user such that the toolbar can cater to both inexperienced and experienced users. Allow users to tailor frequent actions. Provide alternative means of access and operation for users who differ from the "average" user (e.g., physical or cognitive ability, culture, language, etc.; Table 15).

A.9 Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility (Table 16).

A.10 Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search,

Table 11 Consistency and standards checklist

#	Review checklist	Yes No N/A	Comments
4.1	Are the locations of buttons on toolbar in the browser fixed and consistent?		
4.2	Is the terminology used consistently?		
4.3	Are icons labeled clearly?		
4.4	Are the titles of popups consistent?		
4.5	Is color tone used on toolbar consistent with browser?		
4.6	Does the menu structure and buttons layout match the task structure?		
4.7	Do online instructions appear in the consistent place?		
4.8	Do toolbar error messages follow hosted browsers' message standards?		
4.9	Do toolbar warnings follow hosted browsers' warning standards?		
4.10	Are attention-getting techniques used with care?		
4.11	Is the attention-getting technique used only for warning of the fraud detection or also for exceptional conditions?		
4.12	Is the most important information placed at the beginning of the prompt?		
4.13	Are advices for users named consistently across all prompts in the toolbar, no matter what the risk level of warning is?		
4.14	Does the structure of menu items' names match their corresponding contents?		
4.15	Do abbreviations follow a simple primary rule?		

Table 12 Help users recognize, diagnose, and recover from errors checklist

#	Review checklist	Yes No N/A Comments
5.1	If toolbar can not verify the legitimacy of web pages, is user kept informed what user could do to check it manually?	
5.2	Is sound used to signal an error?	
5.3	Are popups brief and unambiguous?	
5.4	Are error messages worded so that the toolbar, not the user, takes the blame?	
5.5	Do prompts imply that the user is in control?	
5.6	Are error messages and warnings correct in grammar?	
5.7	Is wording in error messages and warnings in good manner or politely?	
5.8	Are warnings able to show the origin of error?	
5.9	Do warnings inform the user of the error's severity?	
5.10	Do warnings inform the user how to correct the error?	

Table 13 Error prevention checklist

#	Review checklist	Yes No N/A	Comments
6.1	Are menu items logical, distinctive, and mutually exclusive?		
6.2	Are data inputs case-blind whenever possible?		
6.3	Are data inputs type-sensitive?		
6.4	Does the toolbar alert users if they are about to make a potentially serious error?		
6.5	Do fields in data entry screens and dialog boxes contain default values when appropriate?		
6.6	Is there help or instruction for data inputs?		

Table 14 Recognition rather than recall checklist

#	Review checklist	Yes No N/A	Comments
7.1	For question and answer interfaces, are visual cues and white space used to distinguish questions, prompts, instructions, and user input?		
7.2	Are prompts, cues, and messages placed where the eye is likely to be looking on the screen?		
7.3	Have warning or error messages been formatted using white space, justification, and visual cues for easy scanning?		
7.4	Is it easy to find necessary operation for checking the legitimacy?		
7.5	Does the toolbar gray out or delete labels of currently inactive functions?		
7.6	Have items on a dialog or popup been grouped into logical zones, and have headings been used to distinguish between zones?		
7.7	Are significant button or menu item groups identified and highlighted?		
7.8	Does the toolbar provide mapping: that is, are the relationships between controls and actions apparent to the user?		
7.9	Are inactive menu items grayed out or omitted?		
7.10	Are the optional and non-optional settings of toolbar distinguished?		
7.11	Is it easy to find place for changing toolbar related settings?		

Table 15 Flexibility and minimalist design checklist

#	Review checklist	Yes No N/A	Comments
8.1	If toolbar supports both new and experienced users, are multiple levels of warning detail available?		
8.2	Can user customize the filter settings of toolbar?		
8.3	Can user customize the layout of toolbar?		
8.4	If menu lists are short (seven items or fewer), or there are limited buttons (no more than 10) on the toolbar, can users select an item or button by moving the cursor?		
8.5	Do users have the option of either clicking directly on a field (e.g. menu item, input field, and dialog box)or using a keyboard shortcut?		
8.6	Can users set their own default operation for the detected fraud?		
8.7	Can users nullify their default operation for the detected fraud?		
8.8	If users set their own default alert level too low, is there any warning or reminding for this?		

Table 16 Aesthetic and minimalist design checklist

#	Review checklist	Yes No N/A	Comments
9.1	Is only (and all) information essential to decision making displayed on the screen?		
9.2	Are meaningful groups separated properly?		
9.3	Does each group have meaningful title?		
9.4	Are menu items' titles brief, yet long enough to communicate?		
9.5	Do warnings concisely and correctly show the most important information about phishing detection?		
9.6	Can users be misled to other websites not related to phishing and its prevention?		

Table 17 Help and documentation checklist

#	Review checklist	Yes No N/A	Comments
10.1	Is there any user manual, tutorial, or help documentation available?		
10.2	Is there online help available?		
10.3	Is there instant help available?		
10.4	Is there necessary report to toolbar's provider available, when current version of toolbar can not successfully judge the web page?		
10.5	Do instructions or manuals follow the sequence of user actions?		
10.6	Are instructions and other help documentations understandable?		
10.7	Is it easy to search what user wants to know from help documentation?		
10.8	Is the help function visible and easy to find?		
10.9	Does the terminology of help documentations follow the toolbar general design conventions and standards.		
10.10	Is there context-sensitive help?		
10.11	Is it easy to access and return from the help system?		
10.12	Can users resume work where they left off after accessing help?		
10.13	Is the help detailed enough?		
10.14	Is there report to toolbar's provider available, when user can't find answer from existing help documentations?		

Table 18 Skills checklist

#	Review checklist	Yes No N/A	Comments
11.1	Can users learn from toolbar's functions or documentations what is phishing?		
11.2	Can users learn from toolbar's functions or documentations what are common phishing techniques?		
11.3	Can users learn from toolbar's functions or documentations how to identify the fraud WITH toolbar?		
11.4	Can users learn from toolbar's functions or documentations how to prevent the fraud WITHOUT toolbar?		
11.5	Can users learn from toolbar's functions or documentations how to protect their personal, or confidential information?		
11.6	Are there daily, or weekly phishing reports or news?		
11.7	Can users be informed about serious consequences, if users fail to follow the expected security advice?		

Table 19 Pleasurable and respectful interaction with the user checklist

#	Review checklist
12.1	Is each warning labeled properly in terms of its severity?
12.2	Can warning draw users' attention?
12.3	Is warning too overwhelming to disturb users' pleasant browsing?
12.4	Are there many false and undetermined detection warnings?
12.5	Are there any settings to simplify users' phishing detection sequences?
12.6	Are the frequently used function or button put in the most accessible position?

Table 20 Privacy checklist

#	Review checklist	Yes No N/A	Comments
13.1	Are protected areas completely inaccessible?		
13.2	Can protected or confidential areas be accessed with certain passwords?		
13.3	Is this feature effective and successful, or is the toolbar provider reliable enough to protect all of users' personal information submitted? (e.g. e-mail, telephone number, etc.)		

focused on the user's task, list concrete steps to be carried out, and not be too large (Table 17).

A.11 Skills

The toolbar should support, extend, supplement, or enhance the user's skills, background knowledge of anti-phishing —-not replace them (Table 18).

A.12 Pleasurable and respectful interaction with the user

The user's interactions with the toolbar should enhance the quality of her or his browsing experience. The user should be treated with respect. The design should be aesthetically pleasing—with artistic as well as functional value (Table 19).

A.13 Privacy

The toolbar should help the user protect any personal, private or sensitive information- belonging to the user (Table 20).

References

1. Anti-phishing working group (APWG): Phishing attack Trends Report—March 2006 (2006). http://www.antiphishing.org/reports/apwg_report_mar_06.pdf. Cited 9 Nov 2006
2. Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., Mitchell, J.C.: SpoofGuard (2004). http://crypto.stanford.edu/SpoofGuard/. Cited 27 July 2006
3. Downs, J., Holbrook, M., Cranor, L.: Decision strategies and susceptibility to phishing. In: Proceedings of the 2006 symposium On usable privacy and security, pp. 79–90 (2006)
4. Dhamija, R., Tygar, J.D., Hearst, M.: Why phishing works. In: The proceedings of the conference on human factors in computing systems (2006). http://people.deas.harvard.edu/∼rachna/papers/why_phishing_works.pdf. Cited 11 Nov 2006
5. Dinev, T.: Why spoofing is serious internet fraud. Commun. ACM, **49**(10), 76–82 (2006)
6. FBI National Press Office: Web 'Spoofing' Scams Are a Growing Problem. In: Press Release, Washington D.C. (2003) http://www.fbi.gov/pressrel/pressrel03/spoofing072103.htm. Cited 10 Nov 2006
7. Gartner Inc.: Gartner survey shows frequent data security lapses and increased cyber attacks damage consumer trust in online commerce (2005). http://www.gartner.com/press_releases/asset_129754_11.html Cited 22 November 2006
8. Google: Google safe browsing (2006). http://www.google.com/support/firefox/bin/static.py?page=features.html&v=2.0f. Cited 10 Oct 2006
9. Gutmann, P., Grigg, I.: Security usability. Secur. Priv. Mag. IEEE, **3**(4), 56–58 (2005)
10. Jakobsson, M.: Modeling and preventing phishing attacks. In: Phishing panel of financial cryptography (2005). http://www.informatics.indiana.edu/markus/papers/phishing_jakobsson.pdf. Cited 1 Nov 2006
11. Jakobsson, M., Ratkiewicz, J.: Designing ethical phishing experiments: a study of (ROT13) rOnl auction query features. In: Proceedings of the 15th annual World Wide Web conference, pp. 513–522 (2006)
12. Li, L., Helenius, M.: Anti-phishing IEPlug (2006). http://www.cs.uta.fi/∼ll79452/ap.html. Cited 1 Sep 2006
13. Netcraft: Netcraft anti-phishing toolbar (2006). http://toolbar.netcraft.com/. Cited 18 November 2006
14. Nielsen, J.: Heuristic evaluation online writings (1994). http://www.useit.com/papers/heuristic/. Cited 18 October 2006
15. Pierotti, D.: Usability techniques: heuristic evaluation—a system checklist (1998). http://www.stcsig.org/usability/topics/articles/he-checklist.html. Cited 18 October 2006
16. PhishTank: PhishTank—join the fight against phishing (2006). http://www.phishtank.com/. Cited 5 Nov 2006
17. Stop-phishing group (2006). http://www.indiana.edu/∼phishing/?people=external. Cited 20 Oct 2006
18. Wu, M., Miller, R., Garfinkel, S.: Do security toolbars actually prevent phishing attacks? In: Proceedings of the CHI 2006. 22–27 April 2006 Montréal, pp. 601–610 (2006)
19. Zhang, Y., Egelman, S., Cranor, L., Hong, J.: Phinding Phish: evaluating anti-phishing toolbars. In: Carnegie Mellon University, CyLab Technical Report. CMU-CyLab-06-018 (2006). http://www.cylab.cmu.edu/default.aspx?id=2255. Cited 15 Nov 2006