doi:10.1093/comjnl/bxh000

Protecting Users Against Phishing Attacks

Engin Kirda, Christopher Kruegel

Technical University of Vienna
Email: engin@infosys.tuwien.ac.at, chris@auto.tuwien.ac.at

Phishing is a form of online identity theft that aims to steal sensitive information such as online banking passwords and credit card information from users. Phishing scams have been receiving extensive press coverage because such attacks have been escalating in number and sophistication. According to a study by Gartner, 57 million US Internet users have identified the receipt of e-mail linked to phishing scams and about 2 million of them are estimated to have been tricked into giving away sensitive information. This paper presents a novel browser extension, AntiPhish, that aims to protect users against spoofed web site-based phishing attacks. To this end, AntiPhish tracks the sensitive information of a user and generates warnings whenever the user attempts to give away this information to a web site that is considered untrusted.

Received 00 Month 2005; revised 00 Month 2005

1. INTRODUCTION

As far as people with criminal intentions are concerned, identity theft is an old idea. People with malicious intentions often impersonate people that victims intuitively trust to trick them into giving away their money or belongings. An impostor in police uniform, for example, will not cause many victims to become suspicious and they will comply with whatever they are told. Similarly, *phishing* is a form of online identity theft that aims to steal sensitive information from users such as online banking passwords and credit card information.

Phishing attacks use a combination of social engineering and technical spoofing techniques to persuade users into giving away sensitive information (e.g., using a web form on a spoofed web page) that the attacker can then use to make a financial profit.

Phishing scams have been receiving extensive press coverage because such attacks have been escalating in number and sophistication. Many online service providers believe that their reputation is at stake and fear that users will lose confidence in electronic commerce. According to a study by Gartner [1], 57 million US Internet users have identified the receipt of e-mail linked to phishing scams and about 2 million of them are estimated to have been tricked into giving away sensitive information. The phishing problem has become so serious that the German Minister of Internal Affairs recently drew attention to the urgency of the problem and called upon researchers and industry to find solutions [2].

In this article, we present AntiPhish, a browser extension that aims to protect inexperienced users against spoofed web site-based phishing attacks. AntiPhish keeps track of the sensitive information of a user and generates warnings whenever sensitive information is typed into a form on a web site that is considered untrusted. The tool has been implemented as a Mozilla Firefox plug-in and is free for public use.

The article is structured as follows: The next section provides a brief overview of the different types of phishing attacks and discusses a real-world example. Section 3 describes our solution, AntiPhish, and provides details about its implementation. Section 4 presents an example that shows how AntiPhish mitigates a typical phishing attempt. Section 5 discusses related work. Section 6 presents future work and Section 7 concludes the article.

2. TYPES OF PHISHING ATTACKS

In this section, we give a brief overview of the different types of phishing attacks to familiarise the reader with the threat. A real-world phishing attack is presented in Section 2.3.

2.1. Spoofing e-mails and web sites

Phishing attacks fall into several categories. The earliest form of phishing attacks were e-mail-based and

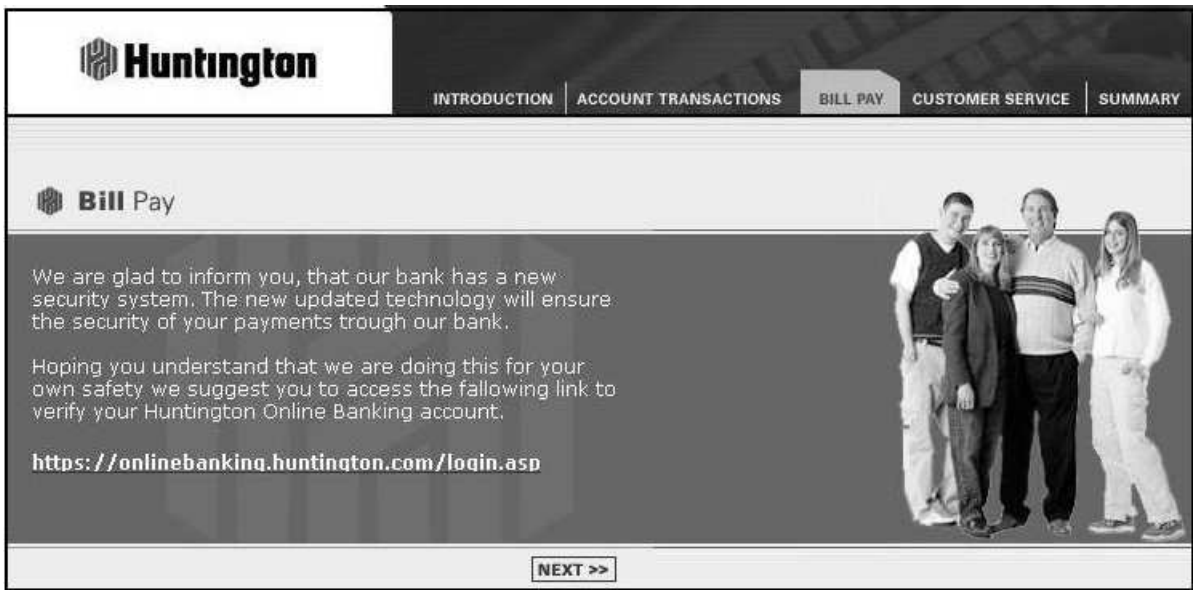

FIGURE 1. Part of a real phishing e-mail that tries to lure the victim into giving away sensitive personal information.

they date back to the mid 90's. These attacks involved spoofed e-mails[1] that were sent to users where attackers tried to persuade the victims to send back their passwords and account information. Although such attacks may be successful today, the success rate from the point of view of the attackers is lower because many users have learned not to send sensitive information via e-mail. A possible reason is that many security-sensitive organisations such as banks do not provide interactive services based on e-mail where the user has to provide a password. Most organisations, obviously, use their web sites for providing interactive services because they can rely on encryption technologies such as SSL. As a result, a typical user would find a request to send sensitive information such as a password via e-mail suspicious (especially considering the fact that many Internet users today receive a large number of spam e-mails from people that they do not know).

Hence, many phishing attacks now rely on a more sophisticated combination of spoofed e-mails and web sites to steal information from victims. Such attacks are the most common form of phishing attacks today. In a typical attack, the attackers send a large number of spoofed e-mails that appear to be coming from a legitimate organisation such as a bank to random users and urge them to update their personal information. The victims are then directed to a web site that is under the control of the attacker. This site looks and feels like the familiar online banking web site and users are asked to enter their personal information. Because the victims are directly interacting with a web site that they believe they know, the success rates of such attacks are much higher than e-mail-only phishing attempts [1].

Besides e-mail, as an alternative form of message delivery, attackers have also started to use instant messaging systems such as ICQ or infrastructures such as Internet Relay Chat (IRC) to try to persuade and direct users to spoofed web sites.

Once the victim follows a spoofed link, in order not to raise suspicion and to present the phishing web site as authentic as possible, attackers are employing various techniques. One example is the use of URLs and host names that are obfuscated and modelled so that they look legitimate to inexperienced users. Another example is the use of real logos and corporate identity elements from the legitimate web site. Some attacks also make use of hidden frames and images as well as Javascript code to control the way the page is rendered by the victim's browser.

2.2. Exploit-based phishing attacks

Some phishing attacks are technically more sophisticated and make use of well-known vulnerabilities in popular web browsers such the Internet Explorer to install malicious software (i.e., malware) that collects sensitive information about the victim. A key logger, for example, might be installed that logs all pressed keys whenever a user visits a certain online banking web site. Another possibility for the attacker could be to change the proxy settings of the user's browser so that all web traffic that the user initiates passes through the attacker's server (to perform a typical man-in-the-middle attack).

Exploit-based phishing attacks are not the focus of our work in this article. To mitigate exploit-based

[1] Taking advantage of the weaknesses in the SMTP protocol, attackers can easily fake the *From* e-mail header and spoof e-mails.

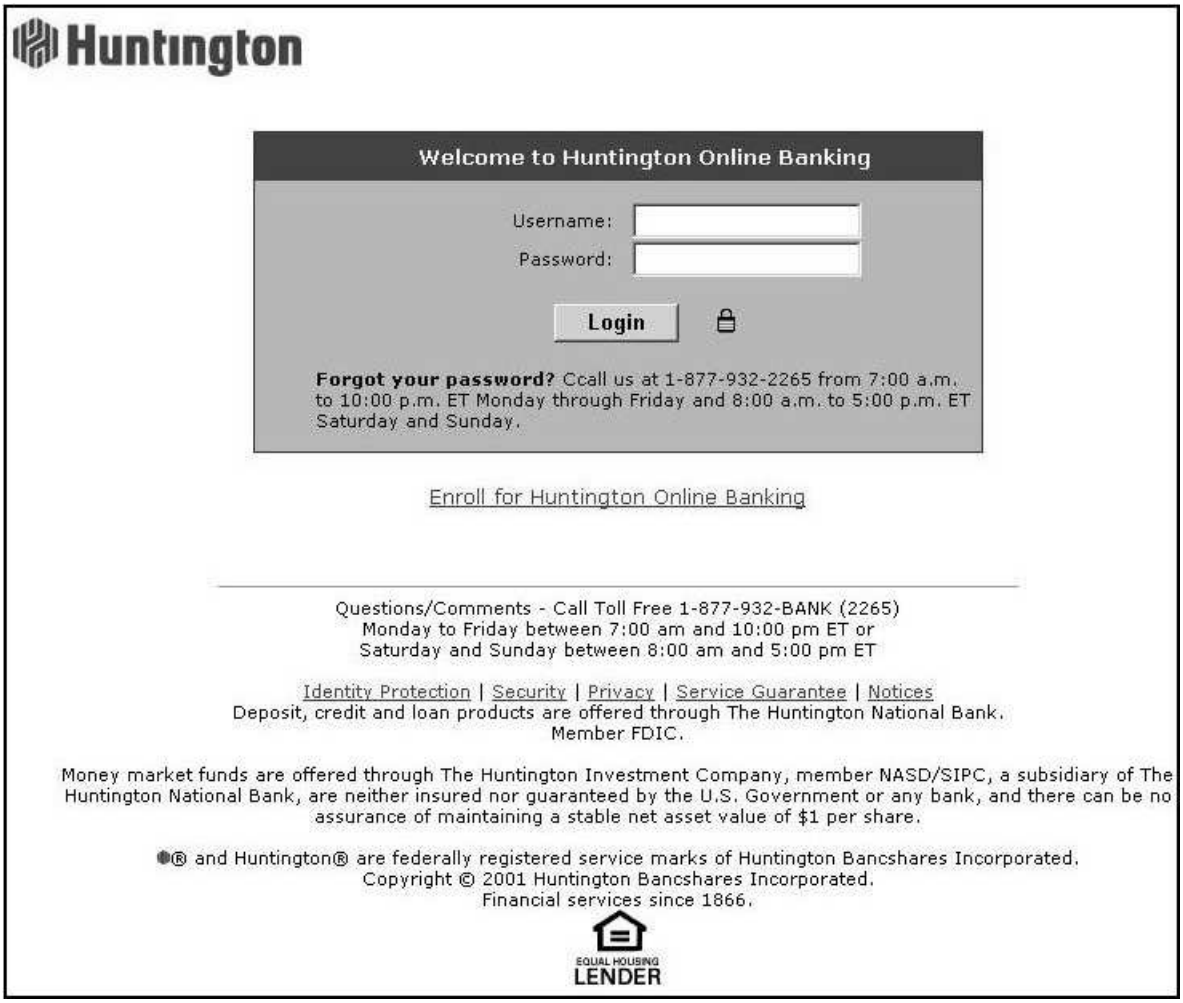

FIGURE 2. Screenshot of the spoofed Hunting online banking page. The login screen closely resembles the legitimate login page [3].

phishing attacks (as well as other security threats that are directly related to browser security such as worms, trojans and spyware), browser manufacturers need to make sure that their software is bug-free and that users are up to date on the latest security fixes. The focus of AntiPhish is to mitigate web site-based phishing attacks that aim to trick victims into giving away their sensitive information.

2.3. A real-world spoofed web site-based phishing attack example

On February 18th 2005, a mass e-mail was sent to thousands of Internet users asking them to verify their Huntington online banking account details. The e-mail claims that the bank has a new security system and that account verification is necessary [4]. The attackers have supposedly inserted a legitimate URL *https://onlinebanking.huntington.com/login.asp* to the bank's online banking web site. However, the link actually points to a spoofed page on the server with the IP address *210.95.56.101*. Figures 1 and 2 show screenshots of the attack. The aim of the attack is to steal the victim's account credentials, credit card information, and personal information such as the social security number. Once the victim enters the requested information, the phishing site redirects to the legitimate bank's web site.

The next section presents our browser extension (i.e., plug-in) for mitigating such phishing attacks.

3. OVERVIEW OF ANTIPHISH

AntiPhish is based on the premise that for inexperienced, technically unsophisticated users, it is better for an application to attempt to check the trustworthiness of a web site on behalf of the user. Unlike a user, an application will not be fooled by obfuscation tricks such as a similar sounding domain name.

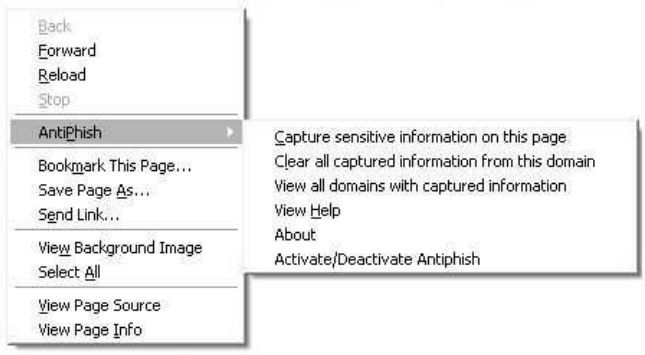

FIGURE 3. The AntiPhish application menu integrated into the browser.

3.1. Main functionality

AntiPhish is an application that is integrated into the web browser. It keeps track of a user's sensitive information (e.g., a password) and prevents this information from being passed to a web site that is not considered "trusted" (i.e., "safe").

The development of AntiPhish was inspired by automated form-filler applications. Most browsers such as Mozilla or the Internet Explorer have integrated functionality that allows form contents to be stored and automatically inserted if the user desires. This content is protected by a *master password*. Once this password is entered by the user, a login form that has previously been saved, for example, will automatically be filled by the browser whenever it is accessed. Antiphish takes this common functionality one step further and tracks *where* this information is sent.

Figure 3 shows the right-click pop-up menu in the browser with the integrated AntiPhish menu items. After AntiPhish is installed, the browser prompts a request for a new master password when the user enters input into a form for the first time. After this password is entered, the AntiPhish menu can be used to capture and store sensitive information. The master password is used to encrypt the sensitive information before it is stored. The symmetric DES algorithm is used for the encryption and decryption.

In our current implementation, user interaction is needed to tell AntiPhish that a piece of information on a page is important and that it should be protected against phishing attempts. After the user enters sensitive information such as a password, the AntiPhish menu is used to scan the page and to capture and store this information. Currently, the contents of all HTML text field elements of type *password* are captured and cached.

Besides storing the sensitive information, AntiPhish also stores a mapping of where this information "belongs" to. That is, the domain of the web site where this information was originally entered is also stored. We use domains instead of web site addresses because some web sites are hosted on multiple servers with different addresses (e.g., the main web site might have the address *www.ba-ca.com* and based on load, the online banking service might be hosted on *online1.ba-ca.com* and *online2.ba-ca.com*). Hence, if web server addresses or URLs are used instead of domains, false phishing alarms could be generated.

In our prototype, we provide simple dialogs for the management of stored sensitive information. The user can see a list of web site domains from which sensitive information has been captured and has the possibility of clearing this cached information[2].

If the user would like to use the *same* piece of sensitive information (e.g., the same password) on multiple web sites, this information has to be captured by AntiPhish for all sites where it is being used. This is typically done by first deactivating AntiPhish from the menu in order to prevent it from generating false phishing alerts.

3.2. Controlling the sensitive information flow

As far as AntiPhish is concerned, every page that contains a form is a potential phishing page. HTML form elements that can be used by the attacker to phish information from the user are text field elements of type *text* and *password* and the HTML text area element. Hence, whenever the user enters information into any of these form elements (e.g., the user presses a key or pastes text), AntiPhish checks the list of previously captured values (i.e., the "watch list"). For each value in this list that is *identical* to the one just entered by the user, the corresponding domain is determined. If the current site is not among these domains, a phishing attempt is assumed. The reason is that sensitive information is about to be transmitted to a site that is not explicitly listed as trusted. If AntiPhish detects, for example, that the user has typed his online banking password into a text field on a web site that is not in the online banking web site domain (i.e., an "untrusted" web site), then it generates an alert and redirects to an information page about phishing attacks.

Interaction events that the user generates within the browser are used to intercept sensitive information flow to untrusted web sites before the user can submit the information. AntiPhish is activated every time the user presses a key, loads a new page, clicks the mouse or has the current focus on a text element (i.e., text field or text area).

The flowchart in Figure 4 depicts how the sensitive information flow is controlled by AntiPhish.

[2]Note that we do not show passwords in clear text form for obvious reasons.

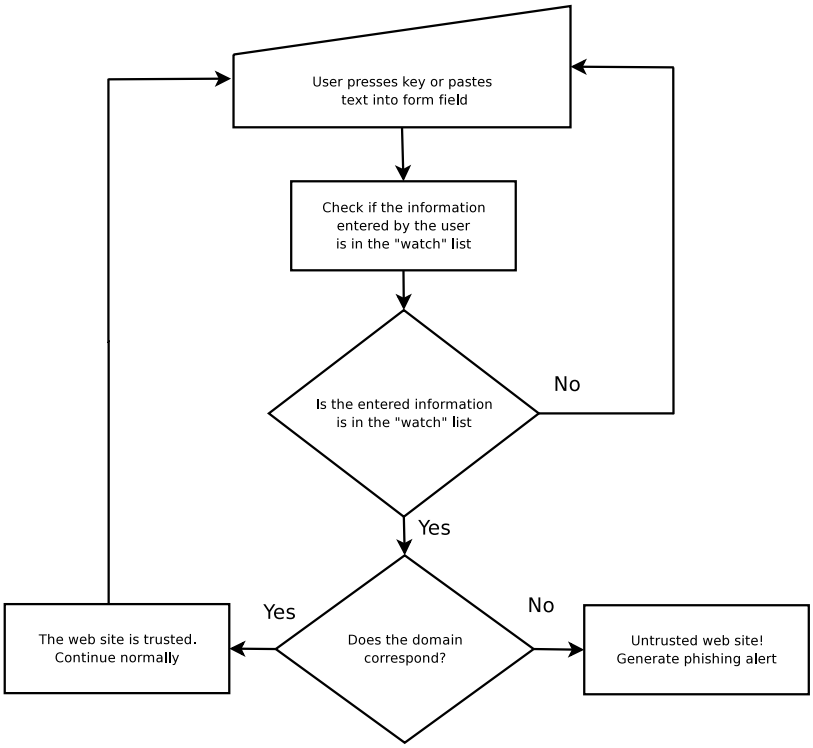

FIGURE 4. Flowchart showing how the sensitive information flow is controlled by AntiPhish

3.3. Possible ways of by-passing AntiPhish with Javascript

As long as the web page that the user is viewing is pure HTML, AntiPhish can easily mitigate phishing attacks. This is because the attacker can only steal the sensitive information in the page after the user performs a submit. Before this can happen, however, AntiPhish detects that sensitive information has been typed into a form and cancels the operation.

Stopping a phishing attack in an HTML page that has Javascript, on the other hand, is not that easy and special care has to be taken. Javascript is a powerful language that is widely used in web pages for providing functionality such as submitting forms, opening windows, intercepting events and performing input validity checks. At the same time, however, Javascript gives the attacker a wide range of possibilities for by-passing a monitoring application such as AntiPhish.

Just as AntiPhish creates hooks for intercepting user generated events such as key strokes, the attacker can also create such hooks using Javascript embedded into the HTML page. Instead of waiting for the user to press a submit button to send the information, the attacker could intercept the keys that are pressed and send the information character by character to a server of her choice. Typically, this is done by modifying the URL of an existing or hidden image to a web site that the attacker controls (e.g., if "a" has been pressed, an image URL may be set to *http://attacker.com/key?a*). Another possibility for the attacker could be to set a simple timer and to capture "snapshots" of the information in the forms. In this way, an important part of the information could be captured without the user ever hitting a submit button.

The easiest solution to the Javascript problem is to deactivate Javascript on a page that contains forms. Unfortunately, this solution is not feasible because, as mentioned before, a large number of web sites use Javascript for validation and submission purposes.

The solution we use in AntiPhish is to *deactivate* Javascript every time the focus is on an HTML text element and to *reactivate* it whenever the focus is lost. Using this technique, we ensure that the attacker is not able to create hooks, timers and intercept browser events such as key presses while the user is typing information into a text field. At the same time, we ensure that the legitimate Javascript functionality on a page (e.g., such as input validation routines) are preserved. By the time the focus is lost from the text element and Javascript is reactivated, AntiPhish has already determined if the information that was typed into the text element is sensitive. If the web site is untrusted, the operation can be cancelled. One side-effect of our approach is that legitimate event-based Javascript functionality such as input validation based on key presses will not function. The use of key press events for input validation, however, is uncommon.

Most web sites perform client-side input validation once before a form is submitted.

3.4. Implementation details

We implemented the prototype of AntiPhish as a Mozilla browser extension (i.e., plug-in). Mozilla browser extensions are written using the Mozilla XML User-Interface language (XUL) [5] and Javascript. The Mozilla implementation of AntiPhish has a small footprint and consists of about 900 lines of Javascript code and 200 lines of XUL user interface code. We used Paul Tero's Javascript DES implementation for safely storing the sensitive information [6].

In the next section, we describe an example scenario where AntiPhish intercepts a phishing attempt.

4. A PHISHING PREVENTION EXAMPLE

Suppose an inexperienced user regularly uses the Bank Austria Creditanstalt (BA-CA) bank's online banking service. The online banking password of this user is a possible phishing target.

The first time this user accesses the online banking web page (*https://online.ba-ca-.com/bach/de/login/index.html*) after the installation of AntiPhish, the password of the user is captured and stored using the integrated menu. This can be either done by the user herself (e.g., with the help of a more experienced user) or by the system administrator.

Now suppose that an attacker has set up a phishing site at the IP address *128.131.172.93* and has sent out thousands of e-mails containing a link to this server. Cloning the bank's login page is trivial: The attacker only needs to save the login page locally[3] using a browser and to minimally adapt the HTML source so that the submitted information is sent to the attacker's server.

The victim (i.e., our inexperienced user) believes that the e-mail is authentic and clicks a link that has been masked as:

```
<a href="http://128.131.172.93">
  http://online2.ba-ca.com
</a>
```

Once the page loads, the victim is presented the BA-CA login page and does not suspect anything to be wrong. When the victim starts typing in her user ID, she is prompted for the master password. Note that a user only has to enter the master password once as long as the browser window is open[4]. The master password is then used to decrypt the stored sensitive information and to use it for input comparison as discussed in Section 3.1.

[3]Many browsers have offline capabilities now so when a page is saved, all images and corporate identity elements within the page are saved as well.

[4]In Mozilla and Internet Explorer, an extension instance is started for every open browser window.

After typing in her user ID, the victim clicks the password text field on the BA-CA login page and starts typing in her password. After the last character of her password is entered, an alert message is displayed telling her that she is a potential victim of a phishing attempt because the sensitive information she used on site *https://online.ba-ca.com* (i.e., the web site domain *ba-ca.com*) is about to be passed to the untrusted site *128.131.172.93* (see Figure 5).

5. RELATED WORK

Two similar, browser-based plug-in solutions exist to mitigate Phishing attacks [7, 8]. Both solutions are from Stanford university. PwdHash [7, 9] is an Internet Explorer plug-in that transparently converts a user's password into a domain-specific password so that the user can safely use the same password on multiple web sites. A side-effect of the tool is some protection from phishing attacks. Because the generated password is domain-specific, the password that is phished is not useful. The problem, however, is that the solution only works for protecting passwords and does not work for sensitive information that is needed in unaltered form by a web site such as credit card information and social security numbers (Note that our prototype implementation of AntiPhish also captures password text field elements. This, however, is an implementation choice and not a conceptual limitation of our approach). SpoofGuard [8, 10] is a plug-in solution specifically developed to mitigate phishing attacks. The main difference between SpoofGuard and AntiPhish is that SpoofGuard is symptom-based. That is, the plug-in looks for "phishing symptoms" such as similar sounding domain names and masked links in the web sites that are visited. Alerts are generated based on the number of symptoms that are detected. AntiPhish, in comparison, is user input-based and guarantees that sensitive information will not be transferred to a web site that is untrusted.

Verisign has recently started to provide an antiphishing service [11]. The company is crawling millions of web pages to identify "clones" in order to detect phishing web sites. Furthermore, black lists of phishing web sites are maintained. AOL has also recently announced that it is planning to integrate black list-based antiphishing support into the Netscape browser [12]. The browser will not allow the user to connect to web sites that are black-listed. The problem with crawling and black listing proposals is that the antiphishing organisations will find themselves in a race against the attackers. This problem is analogous to the problems faced by antivirus and antispam companies. There is always a window of vulnerability during which users are susceptible to attacks. Furthermore, listing approaches are only as effective as the quality of the lists that are maintained.

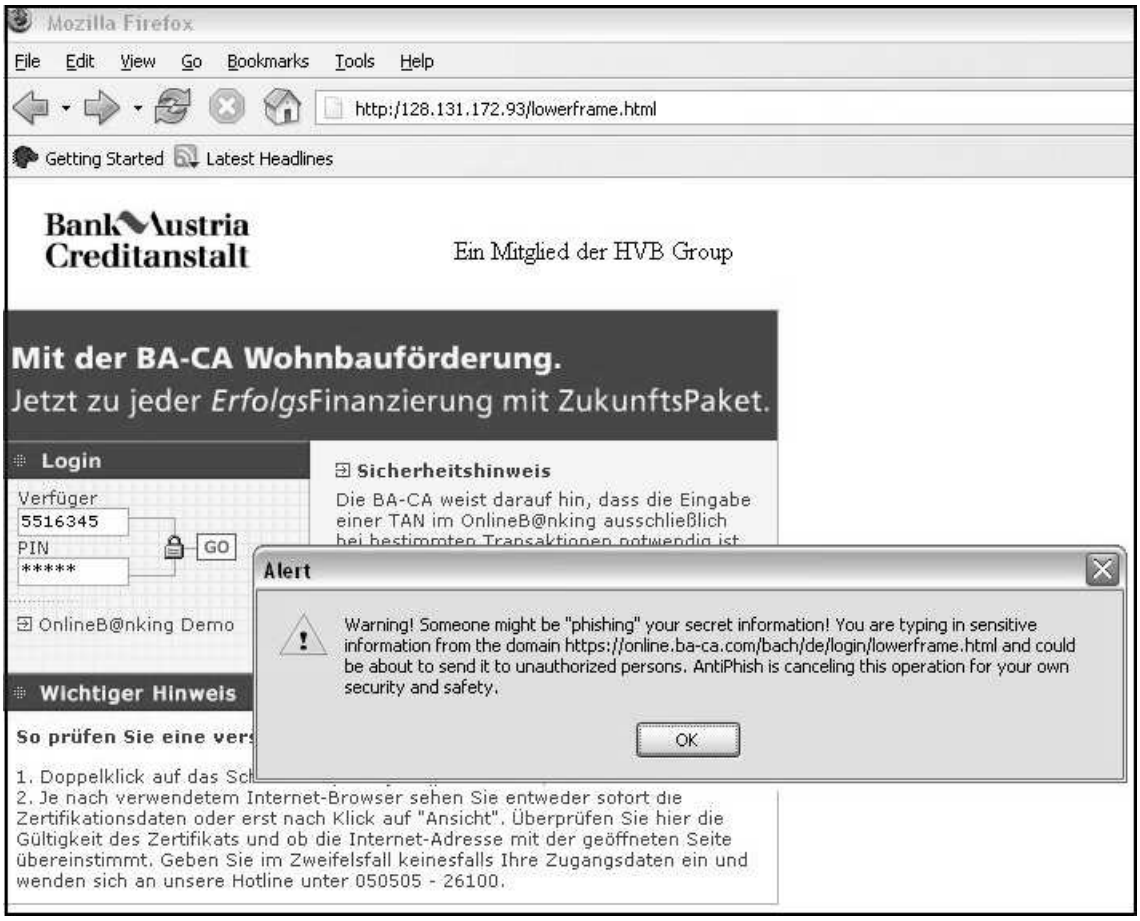

FIGURE 5. The phishing alert message box.

6. FUTURE WORK

We are currently working on the implementation of AntiPhish for the Internet Explorer (IE) browser. Supporting IE is important because a large majority of Internet users are using this browser.

AntiPhish is free for public use. The prototype of AntiPhish can be downloaded from [13]. We are also planning to officially register the project with the Mozilla extensions web site [14].

As discussed in Section 3, AntiPhish currently needs user support to capture and store sensitive information. For some users, it might be better to provide a mode where sensitive information is automatically captured and stored. This could be done by capturing and caching the information every time information is entered and submitted to a web site. In order to implement this functionality, submission events also need to be intercepted. In Mozilla Browsers, submission events are easily captured by implementing call-back functions that are automatically invoked whenever a form is submitted.

7. CONCLUSION

Phishing is a form of online identity theft that aims to steal sensitive information from users such as online banking passwords and credit card information. The last years have brought a dramatic increase in the number and sophistication of such attacks. Although phishing scams have received extensive press coverage, phishing attacks are still successful because of many inexperienced and unsophisticated Internet users. Attackers are employing a large number of technical spoofing tricks such as URL obfuscation and hidden elements to make a phishing web site look authentic to the victims.

The most effective solution to phishing is training users not to blindly follow links to web sites where they have to enter sensitive information such as passwords. However, expecting that all users will understand the phishing threat and surf accordingly is unrealistic. There will always be users that are tricked into visiting a phishing web site. Therefore, it is important for researchers and industry to provide solutions for the phishing threat.

To our knowledge, only two academic phishing solutions have been presented to date and both solutions have limitations. Several companies such as AOL have announced plans to provide some phishing support with their browsers. Most proposed phishing solutions are based on the crawling of web sites to identify "clones" and the maintenance of black lists of phishing web sites. Such solutions, however, require the antiphishing organisations to be much faster than the attackers. Unfortunately, experience has shown such solutions to be partially effective (e.g., the spam e-mail problem has gotten worse).

This article presents a novel browser extension called AntiPhish that aims to protect users against spoofed web site-based phishing attacks. AntiPhish tracks the sensitive information of a user and generates warnings whenever the user attempts to transmit this information to a web site that is considered untrusted.

As the number of phishing scams continues to grow and the costs of the resulting damages increases, we believe that AntiPhish is a step in the right direction and a useful contribution for protecting users against spoofed web site-based phishing attacks.

REFERENCES

[1] Ollman, G. (2004) The Phishing Guide - Understanding and Preventing, White Paper, Next Generation Security Software Ltd.

[2] Heise Security (2005) German Interior Minister Schily requests protection against online scams, http://www.heise.de/security/

[3] The Antiphishing Working Group (2004) Home Page, http://www.anti-phishing.org

[4] The Antiphishing Working Group (2004) Phishing Activity Trends Report, http://www.anti-phishing.org

[5] Dikean, N. (2005) The XULTU Tutorial, http://-www.xulplanet.com/

[6] Tero, P. (2001) Javascript DES, http://-www.shopable.co.uk/des.html

[7] Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J. (2005) A Browser Plug-In Solution to the Unique Password Problem, http://crypto.stanford.edu/PwdHash/

[8] Boneh, D. (2005) SpoofGuard Home Page, http://-crypto.stanford.edu/SpoofGuard/

[9] Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J. (2005) Stronger Password Authentication Using Browser Extensions, 14th Usenix Security Symposium, Baltimore, MD, USA

[10] Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., Mitchell, J. (2005) Client-side defense against web-based identity theft, 11th Annual Network and Distributed System Security Symposium (NDSS '04), San Diego, CA, USA

[11] Verisign Home Page (2005), Anti-Phishing Solution, http://www.verisign.com/verisign-business-solutions/anti-phishing-solutions/

[12] News.Com (2005) Netscape readies antiphishing browser, http://news.com.com/2100-7355_3-5558006.html

[13] Kirda, E., Kruegel, C. (2005) AntiPhish Home Page, http://www.infosys.tuwien.ac.at/antiphish/

[14] Mozilla Extensions (2005) Home Page, http://update-.mozilla.org/extensions/