1 Browser Enhancements for Preventing Phishing Attacks

Cynthia Kuo, Bryan Parno, Adrian Perrig
Carnegie Mellon University

Phishing attacks exploit human vulnerabilities. They play on our feelings of greed, our instinct to help others, or our need to protect what we have. Phishers often use the same social engineering strategies that con artists have used in the offline world for generations. And despite years of consumer education efforts, users continue to be scammed by offline con artists. In fact, in the first half of 2005, victims of telemarketing scams lost an average of $4100 – more than double the average loss in 2004 [7].

The continued "success" of con artists in the offline environment demonstrates the effectiveness of social engineering in manipulating human judgment. Relying on human judgment to combat phishing attacks – which are often social engineering attacks – is a curious choice. But that is exactly what has happened. In many anti-phishing schemes, users are responsible for detecting and avoiding phishing sites.

For example, researchers at Carnegie Mellon developed a browser extension to display *visual hashes* of web site certificates [4, 12, 15]. Each SSL-encrypted web site has a unique digital certificate to authenticate its identity to users. However, certificate verification involves comparing long strings of alphanumeric characters, which is difficult and tedious for human users. A certificate can be more easily verified when it is represented as a picture (i.e., a visual hash [12]). Human users are better able to compare or recognize images than alphanumeric strings.

The visual hashes were embedded in a browser toolbar, as illustrated in Figure 1. Users can check that the same image is displayed each time they visit a web site. However, this system is entirely passive; its effectiveness hinges on users' ability to notice a small change in the user interface. Will users notice if an image is different? And if they do, will they stop using the site as a result?

Over the long term, users may become habituated to the visual hashes and may fail to "see" them. Even worse, users may notice a change in the image but ignore the warning because the phishing attack is so compelling. Passive systems that succeed with conscientious users will fail with inattentive or gullible users.

On their own, average users may also be ill-equipped to identify phishing attacks for a variety of reasons:

- Users habitually dismiss security warnings without reading the messages. (This may be a product of bad software design, but the behavior is hard to unlearn.)

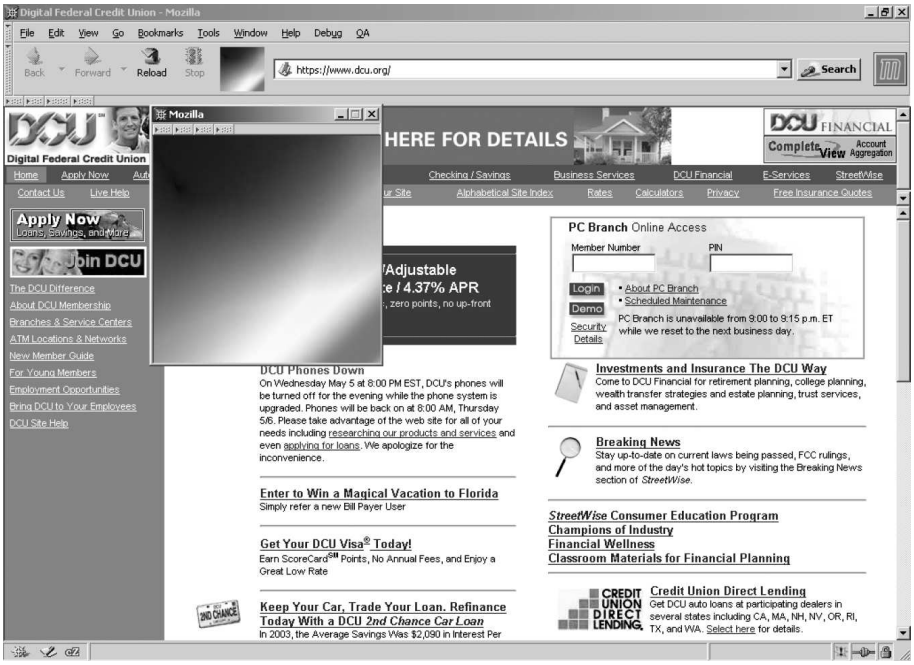


Figure 1: **A Browser Displays the Visual Hash of a Bank's SSL Certificate [15]** *A Mozilla browser extension generates a hash using a server's certificate. This hash is then used to create the visual hash. Since the same visual hash is displayed for all users and does not change over time (unless the certificate changes), it would be possible for phishing sites to spoof the expected image.*

- Users have become accustomed to computers and websites behaving erratically. They often attribute the absence of security indicators to non-malicious errors [16].
- Users may be unfamiliar with the structure of a URL. As of December 2005, many phishing sites still have URLs that look nothing like the legitimate URL (e.g., a phishing site for the Royal Bank of Canada goes to `http://picton.trump.net.au/~wigston/https/...`) [10].
- On a more technical level, most users cannot distinguish between actual hyperlinks and spoofed hyperlinks that display one URL but link to a different URL (i.e., URLs of the form: `<a href=‘‘http://phishing.org/’’> <img src=‘‘ebay-url.jpg’’> </a>`).
- Users are unable to reliably understand domain names or PKI certificates.

All these factors may limit the ability of average users to recognize a phishing attack.

As a result, it is not surprising that users have low confidence in their ability to protect themselves. For example, an October 2005 Consumer Reports survey found that 29% of consumers have cut back on – and 25% have even stopped – shopping online, due to fears over identity theft and fraud [13].

Because the stakes are so high, we should consider technologies that *actively protect* users from making mistakes. The traditional, passive approach – providing users with tools to make informed decisions – may not be sufficient.

1.1 Goals for Anti-phishing Techniques

While most researchers agree on the importance of preventing phishing attacks, few have precisely defined the goals of a technique to effectively combat them. Below, we enumerate these goals, arranged in decreasing order of protection and generality:

1. Ensure that the user's data only goes to the recipient that the user thinks it is going to.
2. Prevent the user's data from reaching an untrustworthy recipient.
3. Prevent an attacker from abusing the user's data.
4. Prevent an attacker from modifying the user's account.
5. Prevent an attacker from viewing the data associated with user's account.

Clearly, an ideal solution would address the first goal (along with the others). However, divining a user's intentions remains a difficult problem, particularly when even the user may find it difficult to quantify his or her precise objectives. The second and third goals, while more constrained than the first, require complete control over the user's data. Although we present techniques to assist with the goal of preventing the user's data from reaching an untrustworthy recipient, ultimately, we cannot guarantee this result; a determined user can *choose* to disclose personal information to an adversary. However, we can guarantee the last two goals via technical measures.

In this section, we discuss two systems that tackle the phishing problem from different angles. The first system uses a browser enhancement that strongly discourages users from submitting information to known phishing pages. This directly addresses the second goal: preventing the user's data from reaching an untrustworthy recipient. As mentioned earlier, we cannot guarantee that this goal will

be achieved; users can deliberately submit their information to a phishing site by overriding our system. However, we expect this to be a rare occurrence. The second system introduces an additional authenticator that a user cannot readily reveal to a malicious party. This ensures that user error alone cannot jeopardize a user's accounts. The system also helps the user avoid phishing sites in the first place, thus guaranteeing that the fourth and fifth goals are met, a result not previously achieved.

1.2 Google Safe Browsing

Google designed a warning system that actively protects users from phishing sites. The team [5] crafted a browser extension that disables the interactive elements on phishing sites. The project faced both technical and user-interface challenges.

On the technical side, it is difficult to automatically detect phishing sites without making classification errors. The ideal system would identify phishing sites as phishing sites and legitimate sites as legitimate sites. Unfortunately, automatic systems typically require a tradeoff between false positives (identifying a legitimate site as a phishing site) and false negatives (failing to identify a phishing site as a phishing site). When the incidence of one type of error goes down, the rate of the other error usually goes up. A browser extension that successfully combats phishing attacks needs to have a false positive rate of zero; otherwise, once users discover that the system incorrectly identifies legitimate sites as phishing sites, they will learn to ignore the warnings.

Initially, we also considered building a broader system for classifying "good," "suspicious," and "bad" sites, but we decided against this scheme for several reasons. First, all sites would want to be categorized as "good." Sites with "bad" labels would want to improve their ratings to increase their legitimacy. These bad sites could spoof the *browser chrome* to get the "good" site ratings. ("Browser chrome" refers to the borders of a browser window. It includes the menus, toolbars, and scroll bars.) Spoofing the browser chrome would be difficult for users to detect. Second, a broad notion of "suspicious" or "bad" is difficult to define. For example, a legitimate business may have extremely customer-unfriendly business practices. Is a site of this type suspicious or bad? Should it be? What qualities would define a site as suspicious? How could identification be accomplished without generating a large number of false positives? For a variety of reasons, in our initial implementations we narrowed our focus to warning users about known blacklisted phishing sites.

On the user interface side, we felt the system must alert users to potential problems without requiring extensive user education – and without annoying the users. We wrestled with several design challenges:

- How do we grab users' attention without annoying them? How do we communicate that the current situation is a dire one, worthy of their attention? How do we differentiate this warning message from all the others?
- How do we convince users to care about a problem they might not understand? What if users have never heard of phishing? What if users don't understand how the Internet works?
- What happens when we don't know whether a site is a legitimate site or a phishing site? How do we encourage users to make an informed decision before using a questionable site?

All these questions reflect the challenges of developing a technique to protect Internet users from phishing attacks. In the following section, we focus exclusively on user interface challenges.

1.2.1 User Interface Challenges

If you spend enough time surfing the web, chances are that you have seen advertisements posing as security warnings, similar to the advertisement in Figure 2. Notice how the ad looks almost identical to a legitimate Windows warning message. Now that users have become accustomed to such ads, they tend to assume that all pop-up warnings are ads. As a result, users have difficulty evaluating the trustworthiness of legitimate security warnings.

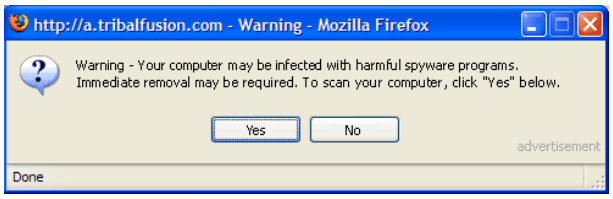


Figure 2: An Advertisement Posing as a Security Warning

In trying to craft a legitimate security warning, we iterated through a number of designs. These designs were tested using a series of informal user studies. User study participants consisted of non-engineering employees who we thought would represent typical Internet users. The participants came from a variety of countries, so many were non-native English speakers. Some were familiar with the concept of phishing, while others were not. Participants were told that we were studying site registration, rather than phishing alerts. When they arrived for the study, participants were presented with a page of links and were told to register at one of the

(randomly selected) sites. When participants navigated to the registration pages, a phishing alert appeared. The experimenter then asked participants to explain their reaction to the alert and their interpretation of the warning message. If multiple designs were being tested, the experimenter then directed participants to different pages. Each page contained a different design. Participant were asked to compare and contrast the current design with the previous one(s). This method yielded some unexpected observations, as we describe below.

In our first attempt at designing a phishing alert, a friendly-looking bubble hovered over the form fields that users should avoid. Initial user studies quickly revealed this was unacceptable:

- Anything that pops up on a web page looks like an ad.
- Users don't read the messages in ads.
- Ads are annoying.

Clearly, users responded badly to the alerts. However, we observed that it was the "floating" aspect of the warning that made it appear less than trustworthy. The warning moved around in the space of the web page, and it was not visually attached to a trusted element. As a result, the origin of the message was unclear. Users had doubts about it. Who is the owner of the message? Should I trust the web page or the warning message? Furthermore, users had difficulty evaluating why one message would be more trustworthy than the other.

The intrusiveness of the bubble also posed a problem. Participants in the user study resented interruptions of their workflow. This was unacceptable for sites that were labeled as only "suspicious." One user commented that the entire Internet is suspicious; warning users to be careful is extraneous. For known phishing sites, participants were generally grateful for the warning, but only once they understood the purpose of the interruption.

In later iterations, we tested different designs for the warning message, varying the color, the shape, and the location of the message. Users objected to the flashier, more eye-catching designs. These designs were regarded as too similar to Internet pop-up ads. Muted colors and simpler layouts were perceived to be more professional.

Crafting the wording of the message may have been the most difficult aspect of the design. How could we give users all the information they need in as little text as possible? Less technically-inclined users were unfamiliar with terms such as "phishing" or "blacklist." More technically-inclined users became confused if we avoided using the technical terms.

In addition to the terminology issues, we encountered conceptual problems as well. One user did not understand *why* she was being alerted. She did not

understand how simply entering a username and password could put her identity at risk. She was careful with her credit card information, but she did not know how providing other information could be dangerous. Other users felt lost without explicit instructions as to what they should do next.

Perhaps our most interesting finding was the importance of giving users a sense of closure. Some users expressed a strong sense of outrage and a desire for retribution. *Even though they had not entered their personal information, they still felt victimized*. Being directed to a phishing page was enough to make users feel duped. One indignant user asked how he could report the page to authorities. We pointed out that the page was a known phishing site; otherwise the system could not have warned him. He then asked why the page had not been taken down. His indignation fueled a desire to take constructive action, such as reporting the site to an authority or trying to prevent other people from being victimized.

1.2.2 Design Principles for an Anti-phishing User Interface

Based on the observations described in Section 1.2.1, we developed the following set of design principles for our anti-phishing warnings.

Assume no prior knowledge of technical terminology. Users ignore many warnings because they do not understand what they mean. Unfamiliar technical terms should be avoided.

Establish the trustworthiness of the message. Visually, it is critical to maintain a professional look and feel. In addition, referring to a trusted entity helps build users' confidence.

Match the intrusiveness of the warning to the severity of the warning. Only interrupt users' workflow when absolutely necessary. In extreme cases, an intrusive mechanism may be acceptable. For less extreme cases, it is important that a mechanism can be easily ignored and dismissed.

Recommend what action(s) to take. Explicitly stating what actions users should take helps to reduce confusion. It is also important to outline the consequences of an action: if users choose to ignore the warning and visit the phishing site anyway, what could happen?

Give users a sense of closure. Phishing evokes strong emotions, including outrage, embarrassment, and a desire to take constructive action. Navigating away from a phishing site may not be a satisfying solution; some users may feel it is the

equivalent of witnessing a crime and walking away. Channeling users' indignation into a constructive outlet provides closure to the incident.

Teach users about the problem. A multi-level interface allows users to access more detailed information if they are curious. Users often appreciate having the ability to look at the information, even if they don't use it.

1.2.3 Final Design of the Anti-phishing User Interface

In our final design, we place a warning icon in the URL address bar, and warning bubbles extend from the icon into the main viewing area of the browser. In Firefox, the SSL lock icon already appears in the URL address bar. The phishing icon appears in the same area, for consistency. The warnings are visually linked to the browser chrome so that users will perceive the messages to be more trustworthy (and less ad-like). The content of the messages also contains the Google logo so that users will know who generates the messages.

When users encounter phishing sites on our blacklist, the entire viewing area of the browser (i.e., where the page is displayed) is grayed out. This change is hard to miss and immediately focuses users' attention on the brighter white warning message. Users intuitively interpret the dark color as indicative of something negative, even without reading the warning. In addition, the magnitude of the change emphasizes the severity of the message. For users who are not paying close attention, we reinforce the message by disabling links and blocking keystroke entry onto the phishing page. We also disable the SSL lock icons so that users will not receive conflicting messages. We found that users do not mind being interrupted if the action is justified. Users consider a known phishing site to be a legitimate reason to interrupt their workflow. If users opt to continue using the phishing site, they can dismiss the warning and interact with the page.

It is important to note that graying out the page and disabling keystrokes are extremely intrusive techniques. These measures are never used unless we are certain the page is a phishing site. For suspected phishing sites or simply questionable ones, the warning messages should be much less intrusive.

For the phishing warning itself, we settled on using "Web Forgery" as a title. Titles such as "Phishing Alert" or "Blacklisted Site" turn off users who are unfamiliar with the terms. These are precisely the users who need the warning the most. The amount of text in the bubble is minimal. For example, "This page is very likely to have been designed to trick users into sharing personal or financial information. Entering any personal information on this page may result in identity theft or other fraud." This is enough to communicate that the site is a phishing site, as well as the potential consequences of the user's actions. We also added three links: one for

The User Experience

Bob just received an email from Citibank. It claims that his account will be suspended until he logs in to the Citibank website and verifies his account information. Bob clicks on the link in the email. His browser navigates to the login page. However, the Citibank page turns dark gray as soon as the page loads. A warning message notifies him that the page is a forgery. He tries to click on the page itself, but the page is no longer interactive. The links do not work, and he cannot enter his login information into the form fields. Bob gets worried and closes his browser window. He then opens up a new browser window and types in the proper URL for citibank.com. He sees a message on the home page detailing a phishing attack that targets Citibank. It is the exact email that Bob received! Bob deletes the message from his Inbox and continues checking his email.

Figure 3: Sample End User Experience for Anti-phishing Warnings

users to access more information about the site and learn about phishing; one for users to report the site to Google; and the last for users to access the phishing site regardless of the danger. The option of reporting the site was added to give users a sense of closure, although there is technically no need for it.

Google Safe Browsing is a browser extension to Firefox, rather than a toolbar. The extension does not appear anywhere on the browser chrome. It requires no attention from the user, only appearing when it actively protects the user. (However, if users want to report a site as a phishing site, there is a option in the browser's menu.) This approach differs from a number of anti-phishing toolbars that are available, such as SpoofGuard [1], SpoofStick [3], Netcraft Anti-Phishing Toolbar [8], or Cloudmark Anti-Fraud Toolbar [2]. These toolbars are constantly displayed in the browser chrome, showing site ratings or domain information related to the user's current site. Some, like Cloudmark, also block users from going to phishing sites. However, many rely on users to rate sites or monitor the toolbar's ratings.

1.3 Phoolproof Phishing Prevention

The anti-phishing warnings outlined above focus on actively preventing users from entering their personal information into a phishing page once they land on it. A complementary approach is discussed below. It introduces a second authentication factor to reduce the impact of user error during a phishing attack. It also tries to prevent users from even reaching a phishing page in the first place.

Our system [11] assumes that users can be trusted to correctly identify sites at which they wish to establish accounts. We justify this assumption on the basis of the following observations. First, phishing attacks generally target users with existing accounts. In other words, phishers attempt to fool a victim with an online account into revealing information that the phishers can use to access that account. Second, users typically exercise greater caution when establishing an account than when using the account or when responding to an urgent notice concerning the account. This results, in part, from the natural analogue of the real world principle of caveat emptor, where consumers are accustomed to exercising caution when selecting the merchants they wish to patronize. However, consumers in the real world are unlikely to encounter a Man-in-the-Middle attack or an imitation store front, and so they have fewer natural defenses when online. Our solution addresses these new threats enabled by the digital marketplace. Our approach is largely orthogonal to existing anti-phishing solutions based on heuristics, and it can be combined with these earlier schemes, particularly to protect the user from a phishing attack during initial account establishment.

1.3.1 Design Principles for a Two-factor Authentication System

Based on the system goals described in Section 1.1, we developed the following set of design principles for a two-factor authentication system that resists user error.

Sidestep the arms race. Many anti-phishing approaches face the same problem as anti-spam solutions: incremental solutions only provoke an ongoing arms race between researchers and adversaries. This typically gives the advantage to the attackers, since researchers are permanently stuck on the defensive. As soon as researchers introduce an improvement, attackers analyze it and develop a new twist on their current attacks that allows them to evade the new defenses. For example, phishers responded to attempts to educate users about the benefits of SSL by spoofing SSL indicators or acquiring bogus or illegitimate certificates [6, 17]. Ultimately, heuristic solutions are bound to be circumvented by the adversaries they seek to thwart. Instead, we need to research fundamental approaches for preventing phishing.

Provide mutual authentication. Most anti-phishing techniques strive to prevent phishing attacks by providing better authentication of the server. However, phishing actually exploits authentication failures on both the client and the server side. Initially, a phishing attack exploits the user's inability to properly authenticate a server before transmitting sensitive data. However, a second authentication failure occurs when the server allows the phisher to use the captured data to log in as the victim. A complete anti-phishing solution must address both of these failures: clients should have strong guarantees that they are communicating with the intended recipient, and servers should have similarly strong guarantees that the client requesting service has a legitimate claim to the accounts it attempts to access.

Reduce reliance on users. We must move towards protocols that reduce human involvement or introduce additional information that cannot readily be revealed. Such mechanisms add security without relying on perfectly correct user behavior, thus bringing security to a larger audience.

Avoid dependence on the browser's interface. The majority of current anti-phishing approaches propose modifications to the browser interface. Unfortunately, the browser interface is inherently insecure and can be easily circumvented by embedded JavaScript applications that mimic the "trusted" browser elements. In fact, researchers have shown mechanisms that imitate a secure SSL web page by

forging security-related elements on the screen [17]. Moreover, browsers voluntarily disclose operating system and browser version information to web servers, facilitating such attacks. Given the complexity of current web browsers and the multitude of attacks, we propose to avoid reliance on browser interfaces.

Forgo network monitoring. A naive approach to phishing prevention might involve monitoring a user's outgoing communication and intercepting sensitive data in transit. Unfortunately, this approach is unlikely to succeed. For example, suppose it is implemented to monitor information transmitted via HTML forms. An obvious response on the attacker's part would be to use a Java applet or another form of dynamic scripting to transmit the user's response. Worse, client-side scripting could easily encrypt the outgoing data to prevent this type of monitoring entirely. In the end, this approach is unlikely to provide a satisfactory solution.

1.4 Final Design of the Two-factor Authentication System

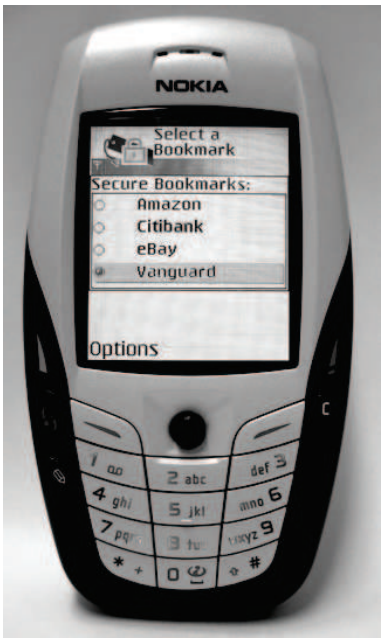
While no automated procedure can provide complete protection, our protocol guards the secrecy and integrity of a user's existing online accounts so that attacks are no more effective than pre-Internet scams (e.g., an attacker may still be able to access a user's account by subverting a company insider). We base our system on the observation that users should be authenticated using an additional authenticator that they cannot readily reveal to malicious parties. Our scheme establishes the additional authenticator on a trusted device, such that an attacker must both compromise the device *and* obtain the user's password to access the user's account.

The trusted device in our system can take the form of a cellphone, a PDA or even a smart watch; we assume the use of a cellphone below. Users cannot readily disclose the authenticator on the cellphone to a third party, and servers will refuse to act on instructions received from someone purporting to be a particular user without presenting the proper authenticator. The system also prevents active Man-in-the-Middle attacks, unlike most other two-factor authentication schemes. Indeed, banks have already reported such attacks against their one-time password systems [9]. In addition, the choice of a cellphone allows us to minimize the effect of hijacked browser windows and facilitates user convenience, since it can be used at multiple machines. We assume that the user can establish a secure connection between his or her cellphone and browser and that the cellphone itself has not been compromised.

To utilize our system, a user must enable it for a new or an existing account. We rely on institutions to implement measures that ensure: 1) their new customers are who they say they are; and 2) the information in existing customers' files is accurate. Institutions have dealt with this problem since well before the existence

The User Experience

Alice has a retirement account at Vanguard. Since all of her retirement savings are accessible online, she worries about the security of her account. Alice contacts Vanguard, which sends a randomly chosen nonce to the physical postal address on file. When Alice receives the nonce in the mail, she logs in to the Vanguard web page and navigates to the cellphone authentication sign-up page. The sign-up page prompts her to enter the nonce into her cellphone. Alice confirms she wants to create a new account on her cellphone, and a bookmark for Vanguard then appears in her phone's list of secure sites. From then on, whenever Alice wants to access her Vanguard account, she navigates to the Vanguard bookmark on her cellphone. The phone directs her browser to the correct website, and Alice enters her username and password to login. After login, the interaction with her retirement account is identical.



Secure Bookmarks: The cellphone displays the secure bookmarks for sites at which the user has established accounts.

Figure 4: Sample End User Experience for System with Login Authenticator

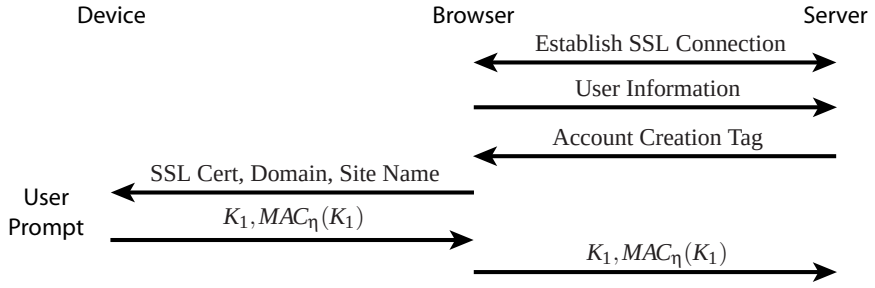


Figure 5: **Account Setup** *Protocol steps for establishing a new user account.*

of computers, and thus, they have well-established techniques for doing so. Using one of these mechanisms, the institution sends a randomly chosen nonce to the user. The user navigates to the institution's website and initiates setup. The setup steps are summarized in Figure 5 and detailed below:

1. The server responds with a specially crafted HTML tag (e.g., `<!-- SECURE-SETUP -->`), which signals the browser that account setup has been initiated.

2. The browser signals the cellphone via Bluetooth, transmitting the server's SSL certificate, domain name, and site name to the phone.

3. The cellphone prompts the user to confirm the creation of the account (to avoid stealth installs by malicious sites) and enter the nonce η provided by the institution. Then, the cellphone creates a public/private key pair $\{K_1, K_1^{-1}\}$ and saves a record associating the pair with the server's certificate. It also creates a *secure bookmark* entry for the site, using the site's name and domain name.

4. The cellphone sends the new public key authenticated with a cryptographic message authentication code (MAC), using the nonce as a key, to the server.

5. The server associates the public key with the user's account, and henceforth, the client must use the protocol described in the next section to access the online account. All other online attempts to access the account will be denied. (This does not preclude Alice from conducting business in person, however.)

Once the user's account has been enabled, the server will refuse to act unless the user is properly authenticated via the established public key pair *and* username/password combination. A user who wishes to access the account must always initiate the connection using the secure bookmark on his or her cellphone. As an alternative, we could have the cellphone detect when a user navigates to a previously registered site. However, a cellphone is ill-equipped to detect if the user visits a phishing site and thus will be unable to prevent the user from disclosing private information to malicious parties. While a phisher would still be unable to access the user's account (without compromising the cellphone), we prefer to help prevent such instances of unnecessary disclosure.

When the user selects a secure bookmark on the cellphone, the cellphone directs the browser to the associated URL. When the remote server provides its SSL certificate, the browser forwards the certificate to the cellphone.

- If the server's certificate matches the certificate that was previously provided, the browser and the server establish an SSL connection. The cellphone assists the browser in performing the client authentication portion of the SSL

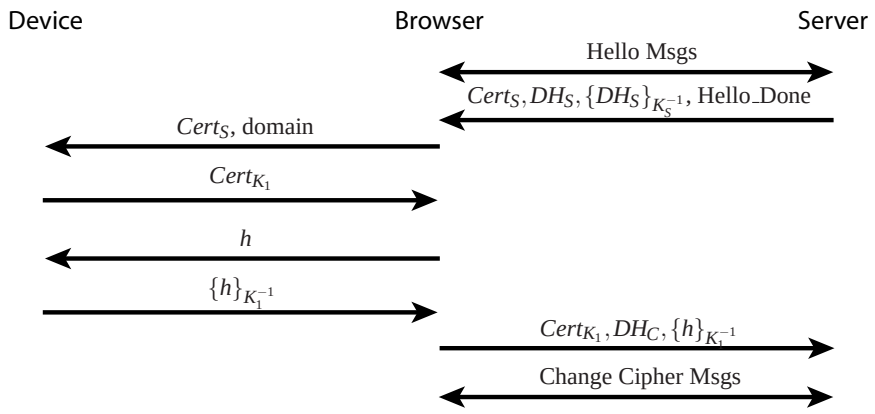


Figure 6: **Secure Connection Establishment** *The browser establishes an SSL connection to the server using client authentication, with help from the cellphone. DH_S and DH_C represent the Diffie-Hellman key material for the server and client respectively, and h is a secure MAC of the handshake messages.*

establishment, using the public key pair associated with this site (the SSL protocol includes a provision for user authentication, but this is rarely used today). The successful establishment of secure connection is illustrated in Figure 6. Once the user has been authenticated and the SSL connection has been established, the user can use the browser to conduct transactions and account inquiries as usual.

- If the certificate check fails, the cellphone closes the browser window and displays a warning message to the user.
- If the server is updating its certificate, then it sends the new certificate along with a signature using the previous key. Upon successful verification, the cellphone can update the certificate it has stored.

Note that we do not change the SSL/TLS protocol; we merely use the cellphone to help the browser establish a session key with the server.

The largest vulnerability in our system arises during account setup (or re-establishment), since the user must ensure that the account is created at a legitimate site. The server also faces an authentication problem, since it must ensure that the person creating the account is the person described by the user information submitted. As discussed earlier, the user's precautions are at a peak during account setup, and we can assist the user with existing heuristics for detecting a spoofed site.

Once the account has been established, the server will not take any action without authenticating the user through the user's private key. Thus, even if the user

is tricked into revealing private information to a phisher or a social engineer, the attacker still cannot access the user's account. Standard keyloggers will also be ineffective, since they can only capture user input, not the private key stored on the cellphone. By storing the user's public key, the server prevents a Man-in-the-Middle attack, since the attacker will not be able to attack the authenticated Diffie-Hellman values from the ephemeral Diffie-Hellman exchange.

The use of secure bookmarks provides the user with a higher degree of server authentication and helps to protect the user from inadvertently arriving at a phishing site, via either a spoofed or a mistyped URL. In addition, we would like to prevent the inconvenience to the user of, for example, making a bank transfer through a phishing website only to discover later that the transaction has not actually taken place. By checking the certificate provided by the server against the stored certificate, the cellphone even protects the user from DNS poisoning and domain hijacking. Our scheme provides very strong guarantees of authenticity to both the client and the server, and thus stops virtually all forms of phishing, DNS spoofing and *pharming* attacks.

Equipping a server with our system requires very minimal changes, namely changes to two configuration options and the addition of two Perl scripts. From the server's perspective, our scheme requires no changes to the SSL protocol. Indeed, most major web servers, including Apache-SSL, Apache+mod_ssl and Microsoft's IIS, already include an option for performing client authentication.

On the client side, we developed an extension to Firefox, an open-source web browser, to detect account creation. We implemented the prototype as a Java *MIDlet* on a Nokia 6630 cellphone. (A Java *MIDlet* is an application that conforms to the Mobile Information Device Profile (MIDP) standard.) Since key generation can require a minute or two, we precompute keys when the user first starts the application, rather than waiting until an account has been created. When the cellphone receives an account creation packet from the browser extension, it selects an unused key pair, assigns it to the server information provided by the browser extension, and then sends the key pair and the appropriate revocation messages to the browser extension. When the user selects a secure bookmark (see Figure 4), the cellphone sends the appropriate address to the browser extension. It also computes the appropriate signatures during the SSL exchange. In practice, we found that account creation on the phone induced a negligible delay, and even the SSL computation required less than two seconds on average, indicating that our system provides a realistic defense against phishing attacks.

This system shares some similarities with the SecurID two-factor authentication system [14]. For example, both systems can be implemented using cell phones or PDAs as trusted devices. However, there are differences as well. In SecurID, both the server and the trusted device generate a new numeric code every 60 sec-

onds. The user must enter the code in a web form and submit it to the server to show that she possesses the trusted device, but there is no server authentication on the user's part. In addition, the system is vulnerable to an active Man-in-the-Middle attack, since a phisher can intercept the value from the user and then use it to access the user's account. As mentioned earlier, similar attacks have already been launched against one-time password systems. In our Phoolproof system, servers and clients authenticate one another. The server's certificate must match a previously provided certificate, and the client must show it possesses the proper key. In addition, Phoolproof uses a bookmark on the cellphone, which directs the browser to the correct website. Finally, since the cellphone participates in the SSL key establishment, the system is not vulnerable to active Man-in-the-Middle attacks.

1.5 Conclusion

Phishing attacks continue to grow increasingly sophisticated. As a result, users are no longer able to differentiate between messages which are legitimate and those which are fraudulent. Because phishing attacks are often social engineering attacks, we believe that technology may be the best counterattack.

We discuss two technologies that actively protect users from phishing attacks. The first is a browser enhancement that warns users when they navigate to a blacklisted phishing site. The second is a system that introduces a second authentication factor for logging in to participating websites. This system also tries to prevent users from ever reaching a phishing page.

The two technologies tackle the phishing problem from different viewpoints: one after the user has reached a phishing site, and the other after the user has established a relationship with a trusted site. These approaches are complementary and can be used in conjunction with one another. Other approaches outlined in this book address facets of the phishing problem that we do not consider here. We encourage you to consider how the different techniques could be used together.

References

[1] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. C. Mitchell. Client-side defense against web-based identity theft. 2004. 11th Annual Network and Distributed System Security Symposium (NDSS '04).

[2] Cloudmark. Cloudmark Anti-Fraud Toolbar. `http://www.cloudmark.com/desktop/ie-toolbar/`.

[3] Core Street. Spoofstick. `http://www.spoofstick.com/`.

[4] R. Dabas, A. Perrig, G. Sinha, T.-F. Yen, C.-H. Yang, and D. Song. Browser enhancement against phishing attacks. Poster at Symposium On Usable Privacy and Security (SOUPS), July 2005.

[5] C. Kuo, F. Schneider, C. Jackson, D. Mountain, and T. Winograd. Google Safe Browsing. Project at Google, Inc., June–August 2005.

[6] Microsoft. Erroneous VeriSign-issued digital certificates pose spoofing hazard. `http://www.microsoft.com/technet/security/bulletin/MS01-017.mspx`, 2001.

[7] National Fraud Information Center. Telemarketing Scams: January - June 2005. `http://www.fraud.org/telemarketing/tele_scam_halfyear_2005.pdf`, 2005.

[8] Netcraft. Netcraft Anti-Phishing Toolbar. `http://toolbar.netcraft.com/`.

[9] Out-law.com. Phishing attack targets one-time passwords. `http://www.theregister.co.uk/2005/10/12/outlaw_phishing/`, Oct. 2005.

[10] Oxford Information Services Ltd. Scam report. `http://www.millersmiles.co.uk/report/1722`, Dec. 2005.

[11] B. Parno, C. Kuo, and A. Perrig. Phoolproof phishing prevention. Proceedings of International Conference on Financial Cryptograpy and Data Security, Feb. 2006.

[12] A. Perrig and D. Song. Hash visualization: A new technique to improve real-world security. In *Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC)*, July 1999.

[13] Princeton Survey Research Associates International. Leap of Faith: Using the Internet Despite the Dangers (Results of a National Survey of Internet Users for Consumer Reports WebWatch). `http://www.consumerwebwatch.org/pdfs/princeton.pdf`, Oct. 2005.

[14] RSA Security. RSA SecurID Authentication. `https://www.rsasecurity.com/node.asp?id=1156`, 2005.

[15] H. E. Tay. Visual validation of SSL certificates in the Mozilla browser using hash images, May 2004. Undergraduate Honors Thesis, School of Computer Science, Carnegie Mellon University.

[16] M. Wu, S. Garfinkel, and R. Miller. Users are not dependable - how to make security indicators to better protect them. Talk presented at the Workshop for Trustworthy Interfaces for Passwords and Personal Information, June 2005.

[17] E. Ye and S. Smith. Trusted paths for browsers. In *Proceedings of the 11th USENIX Security Symposium*. USENIX, Aug. 2002.