In Human Interactive Proofs: Second International Workshop (HIP 2005),
eds. H. Baird and D. Lopresti, Springer, May 2005, pp. 127-141

Phish and HIPs: Human Interactive Proofs to Detect Phishing Attacks

Rachna Dhamija and J.D. Tygar*

University of California, Berkeley
Berkeley, CA 94720
{rachna@sims, tygar@cs}.berkeley.edu

Abstract. In this paper, we propose a new class of Human Interactive Proofs (HIPs) that allow a human to distinguish one computer from another. Unlike traditional HIPs, where the computer issues a challenge to the user over a network, in this case, the user issues a challenge to the computer. This type of HIP can be used to detect phishing attacks, in which websites are spoofed in order to trick users into revealing private information.

We define five properties of an ideal HIP to detect phishing attacks. Using these properties, we evaluate existing and proposed anti-phishing schemes to discover their benefits and weaknesses.

We review a new anti-phishing proposal, Dynamic Security Skins (DSS), and show that it meets the HIP criteria. Our goal is to allow a remote server to prove its identity in a way that is easy for a human user to verify and hard for an attacker to spoof. In our scheme, the web server presents its proof in the form of an image that is unique for each user and each transaction. To authenticate the server, the user can visually verify that the image presented by the server matches a reference image presented by the browser.

1 Introduction

Human Interactive Proofs (HIPs) allow a computer to distinguish a specific class of humans over a network. HIPs can be designed to distinguish a human from a computer, one class of humans from another or one particular human from another human. To do this, the computer presents a challenge that must be easy for that class of humans to pass, yet hard for non-members to pass. Additionally, the results must be verifiable by a computer, and the protocol must be publicly available. [1]

In this paper, we propose a new class of HIPs that allow a human to distinguish one computer, or computer generated message, from another. Unlike

* The authors gratefully acknowledge partial support for this work from the National Science Foundation and the Unites States Postal Service. The views expressed in this work are solely those of the authors and do not necessarily reflect the views of the funding sponsors.

H.S. Baird and D.P. Lopresti (Eds.): HIP 2005, LNCS 3517, pp. 127–141, 2005.

traditional HIPs, where the computer issues a challenge to the user over a network, in this case, the user issues a challenge to the computer. The challenge must:

1. be easy for a particular class of computers to pass,
2. be hard for other computers to pass, even after observing a number of successful authentications,
3. produce results that are easy for a human to verify,
4. use a protocol that is publicly available, and
5. not require the user to have specialized tools.

This class of HIPs can be used by a human to distinguish a known and legitimate website from an unknown one. Such a HIP would be useful in helping humans to detect phishing attacks. In a phishing attack, the attacker spoofs a website (e.g., a financial services website). The attacker draws a victim to a rogue website, sometimes by embedding a link in email and encouraging the user to click on the link. The rogue website usually looks exactly like a known website, sharing logos and images, but the rogue website serves only to capture the users personal information. Many phishing attacks seek to gain credit card information, account numbers, usernames and passwords that enable the attacker to perpetrate fraud and identity theft.

About two million users gave information to spoofed websites resulting in direct losses of $1.2 billion for U.S. Banks and card issuers in 2003, [2] Some phishing attacks have been able to convince up to 5 percent of their recipients to respond and provide sensitive information. [3] Phishing attacks are successful because current web authentication mechanisms do not meet our third requirement; it is not easy for humans to verify that a successful authentication has taken place.

In Section 2, we present an analysis of authentication and anti-phishing schemes using the HIP criteria. In Section 3, we review a system, Dynamic Security Skins (DSS), that meets the HIP criteria and that allows a remote server to prove its identity by displaying an image to the user. [4] We present our conclusions and future work in Section 4.

2 Analysis of Authentication and Anti-phishing Schemes

2.1 Overview

In this section, we analyze authentication and anti-phishing schemes using the criteria for an ideal HIP. In this class of HIP, the user issues a challenge to the computer. The challenge must:

1. Be easy for a particular class of computers to pass. A scheme can meet this criteria if a specified server can reliably authenticate itself to the user without extraordinary resources.

2. Be hard for other computers to pass. Schemes meet this criteria if it is difficult for illigitimate computers to masquerade as the legitimate server, even after observing a number of successful authentications. Furthermore, it should be hard for illegitimate servers to spoof the indicators of a successful authentication.
3. Produce results that are easy for a human to verify. A user should be able to verify that a successful authentication has taken place, without any undue burden on the user in terms of effort, memory or time. Furthermore, in order to meet this criteria, it must by easy for a user to distinguish the legitimate server from a spoofed server.
4. Use a protocol that is publicly available. We note whether the protocol, technical details, source code or policies of the scheme are publicly available.

5. Not require the user to have specialized tools. We note whether the scheme requires users to have any specialized devices or to store or manage any secrets (e.g., cryptographic keys) in order to successfully verify a legitimate server.

In general, attempts to solve the phishing problem can be divided into three categories: third party certification, direct authentication, and phishing specific tools. In order to discover the benefits and weaknesses of each solution, we analyze them using the HIP criteria discussed above. We discuss our analysis in the next sections, and our results are summarized in Table 1.

2.2 Third Party Certification

Hierarchical Trust Models

SSL/TLS. Public Key Infrastructure (PKI) has long been proposed as a method for users and servers to authenticate each other. In PKI, chains of Certificate Authorities (CAs) vouch for identity by binding a public key to an entity in a digital certificate. The Secure Sockets Layer (SSL) protocol and Transport Layer Security (TLS), its successor, both rely on PKI. SSL/TLS allows a server and client to authenticate each other and to negotiate an encryption algorithm in order to communicate privately.

In the typical use of SSL today, only the server is authenticated, by obtaining an SSL server certificate that is signed by a trusted CA. SSL also supports mutual authentication, where both the client and the server are authenticated, however this mode of operation requires the user to obtain a personal certificate. Though it is an active area of research, there is currently no practical scheme for widely deploying signed personal certificates. A further challenge is how to handle the revocation of credentials. SSL is designed to prevent eavesdropping, tampering, and message forgery in client/server communications. Instead of attacking the protocol, most phishing attacks use very simple spoofing techniques to trick users into believing that their connection is "secure". Some phishing attacks exploit the fact that users can not reliably parse domain names (e.g. they can not

distinguish www.paypal.com from www.paypai.com or www.paypal-members-security.com). Many users can not distinguish a legitimate indicator of a secured webpage (e.g. an SSL closed lock icon in the status bar of the browser) from an image of that indicator within the content of a webpage. In many browsers, there is no indicator for "unsecured" sites. This reduces the chance that users will notice spoofed trust indicators when they are inserted into an untrusted page. Other attacks simultaneously display legitimate and illegitimate webpages (e.g., in frames, multiple windows or borderless pop-up windows) in order to trick users into believing that both pages originate from the same website.

It also difficult for users to understand and verify SSL server certificates. Some phishers have gone through the effort of registering a real SSL certificate for their rogue phishing sites that have a similar name to a legitimate site. [5] In order to detect this attack, users must be able to inspect the certificate and to distinguish the domain name of the real website from the rogue site. There are other examples where the CA certificate issuing process has been subverted (e.g., Verisign issued two Class 3 code-signing certificates to an individual who fraudulently claimed to be a Microsoft employee [6]. A user would not be able to detect this attack by inspecting the certificate).

The SSL protocol is publicly available, and the only tool required by the user to authenticate a server is a browser that supports SSL. (To authenticate himself to the server, the user must acquire a personal certificate).

Trustbar. The Trustbar proposal is a third party certification approach that requires website logos to be certified. The authors suggest creating a Trusted Credentials Area (TCA) as a fixed part of the browser window. [7] This area can be used to present credentials from the website, such as logos, icons and seals of the brand that have been certified by trusted certificate authorities or by peers using a PGP web of trust.

The proposal does not specify who will certify logos or how disputes will be resolved in the case of similar logos. If the credentials are signed by trusted CAs, many of the problems of the SSL certificate validation process also apply to this proposal. A strength of this solution is that it does not rely on complex security indicators. However, we expect that all of the spoofing attacks that are common with SSL today will also be applied to the Trustes Credential Area. Because the logos do not change, they can be easily copied and the TCA can be spoofed. An attacker can present an image of the TCA, with the correct logos, in an untrusted page to make it appear legitimate. The success of this attack will depend on the design of the TCA. If there is no visible indicator for unsecured windows, a spoofed TCA may not be detected by users. We expect that phishers will attempt to register logos that can be confused with legitimate logos. Therefore, the strength of this proposal will depend on the strength of the credentials registration process.

The only tool that is required by a user in this scheme is a browser that supports the Trusted Credentials Area.

Table 1. Analysis of authentication and anti-phishing schemes using HIP criteria

Scheme	Easy for specific computer?	Hard for other computers?	Easy for humans to verify?	Protocol available?	Tools required?
SSL	Yes	No	No[1]	Yes	modified browser
Trustbar	Yes	No	No[1]	Yes	modified browser
PGP	Yes	Yes	No[2]	Yes	PGP client/plug-in
3rd Party Seals	Yes	No	No[1]	No	No
AOL Passcode	Yes	No	No[3]	No	SecurID device
SMS Passwords	Yes	No	No[3]	No	cell phone SMS
Passmark	Yes	No	No[4]	No	secure cookie
SRD	Yes	Yes	No[5]	Yes	modified browser
YURL	Yes	No	No[4]	Yes	modified browser
eBay Toolbar	Yes	No	No[1]	No	modified browser
SpoofGuard	Yes	No	No[1]	Yes	modified browser
Spoofstick	Yes	No	No[1]	No	modified browser
DSS	Yes	Yes	Yes[6]	Yes	modified browser

Distributed Trust Models and Third Party Seals

PGP. Another third party approach is the distributed trust model, such as that used by Pretty Good Privacy (PGP). [8] PGP relies on third parties to sign public keys in order to attest that a public key belongs to a particular identity. Unlike centralized PKI schemes, the "web of trust" model relies on individual users to make trust judgments. This allows for more flexibility in how authentication decisions are made, but it requires a great deal of effort on the part of the user to carefully manage keys and to understand the delegation of trust.

It is difficult to break the encryption algorithms, however one simple attack is to create a PGP key using a spoofed identity. Before attesting to a key, users are required to verify the identity of the keyholder (e.g. in a face to face meeting). If users do not take this step before signing a key, an attacker may be able to forge a public key in someone else's name.

Open versions of the PGP protocol are available. In order to authenticate and to verify other parties, the user must have a PGP client application (or a plug-in that implements PGP functionality, which is available for many popular e-mail applications). The user must also store and manage his own private/public key pair and the public keys of others.

[1] This scheme is easy to use, but it is hard to distinguish legitimate sites from spoofs.
[2] This scheme is difficult to use.
[3] This scheme requires users to carry a device and does not help users to distinguish legitimate sites from spoofs.
[4] This scheme requires per-site customization by the user.
[5] This scheme employs blinking border windows that may be distracting to users.
[6] See Section 3.

Third Party Seals. Third party seal programs allow one party to certify another party and offer a "seal of approval" that represents this certification. For example, Verisign allows parties that have purchased a Verisign SSL certificate to post a "Secured Seal" on their websites. [9] Visitors can click on the seal to view a VeriSign-generated pop-up window that contains information about the website's SSL certificate and identity. Phishers spoof this seal by copying the image into their own rogue websites. Some phishers also simulate the pop-window by hosting it on their own server, and many users can not detect that window does not originate from Verisign. While many users can recognize the seal, only sophisticated users can distinguish a legitimate seal from an illegitimate one. These weakness also apply to other third party seal programs like TRUSTe. [10]

The criteria for obtaining the seals is publicly available, and no specialized tools are required by the user.

2.3 Direct Authentication

Direct authentication approaches include user authentication, server authentication and mutual authentication schemes.

Multi-factor User Authentication Multi-factor user authentication schemes use a combination of factors to authenticate the user. The factors are "something you know" (e.g., a password or PIN), "something you have" (e.g., a token or key) or "something you are" (e.g., biometrics).

Passcode. America Online's Passcode program has been proposed as a phishing defense. [11, 12] This program distributes RSA SecurID devices to AOL members. The SecureID device generates and displays a unique six-digit numeric code every 60 seconds. To login to the AOL website, the user enters his password and the SecurID code as a secondary password.

This scheme does reduce the value of collecting passwords for attackers because the passwords can not be used for another transaction. It does not, however, prevent a man-in-the-middle (MITM) attack. In this case, the attacker lures a user to a spoofed AOL website to collect both the primary and secondary passwords. The attacker can immediately present the passwords to the AOL website in order to masquerade as the user. The Passcode program does raise the bar for phishing attacks today, because it requires the phishers to immediately use the passwords they collect. However, we expect that if the bar is raised everywhere, this type of live MITM attack will become common.

Because the server is not authenticated, it is difficult for the user to verify if he is interacting with the AOL webpage or a spoofed webpage. There is also no way for AOL to verify if the passwords it receives are from a legitimate user or from an active MITM attack.

The SecurID protocol is not openly available, and this scheme requires the user to carry a SecurID device.

SMS Secondary Passwords. In another two-factor user authentication scheme, a bank delivers a secondary password to the user's cell phone via Simple Messaging Service (SMS).[13] In order to login and to authorize financial transactions, the user must possess his password and the SMS password. Like the AOL passcode scheme, this scheme is designed to protect the server from fraud, rather than protecting the user from phishing attacks. Because the server is not authenticated, it is difficult for the user to determine if he is interacting with a legitimate bank website or a spoofed website.

The security details of this scheme are not openly available, and this scheme requires the user to carry a cell phone that can receive SMS messages.

Server Authentication Using Shared Secrets

Passmark and Verified by Visa. Shared-secret schemes have been proposed as one approach to prevent phishing attacks. In proposals such as Passmark [14] and Verified by Visa [15], the user provides the server with a shared secret, such as an image or passphrase, in addition to his regular password. The server presents the user with this shared secret, and the user is asked to recognize it before providing the server with his password.

The most obvious weakness of this scheme is that the bank must display the shared secret in order to authenticate itself to the user. If the secret is observed or captured, the image can be replayed until the user notices and changes it.

In the Passmark scheme, the bank server places a secure cookie on the user machine, which must be presented at login. This prevents a classic man-in-the middle (MITM) attack where an attacker interposes himself between the client and the bank. However, a new type of phishing attack is emerging. In this attack, the phisher directs the user to a rogue website, and the users browser opens two windows. The first window displays the real login page of the legitimate bank with legitimate trust indicators (e.g. SSL closed-lock icon, shared secret passmark). The second window displays a webpage from a rogue server. By careful placement of the window, an attacker can convince the user to supply his password. A user may believe that the trust indicators in the first window also apply to the second window, or the user may not even notice that a second window exists.

There is a much easier way to trick the user into revealing his password and passmark. This involves spoofing the Passmark re-registration process. If the user wishes to login using a new browser or a new computer, or if the secure cookie has been deleted, the user must re-register his passmark. Here, the user is shown a "passmark not shown" screen and must enter his password in order to register. This process is inconvenient and also creates a spoofing vulnerabilty. An attacker can direct users to a screen that *claims* that the cookie has been deleted or does not exist. The legitimate Passmark error page asks users to ensure that they have reached the error page by typing in the URL by hand, but a spoofed error page will not include this warning. Spoofing requires no knowledge of the user and requires no special skills other than sending email and creating a website.

A number of attacks are possible that require more difficulty (e.g., breaking the secure cookie, physical observation of the secret image, discovering the potential range of images and then guessing the image). Spoofing is likely to require the least amount of effort to defeat the most people, and we expect that this type of spoofing attack will become common if systems like Passmark are widely deployed.

A final vulnerability of these schemes is that it requires the user to customize each site he wishes to authenticate. The user must be able to recognize the shared secret and associate it with the correct server. Research suggests that users are able to correctly recognize a large number of images. [16] However, if a user is required to remember different images or passphrases for a number of different servers, any difficulty in recognizing an image can be exploited by an attacker.

The security details of the Passmark scheme are not publicly available. This scheme requires the user to store a secure cookie in his browser.

Server Authentication Using Self-shared Secrets These server authentications schemes differ from shared-secret schemes, because an additional secret is shared only with the user's device, rather than shared with a remote server.

Synchronized Random Dynamic Boundaries. Ye and Smith propose "Synchronized Random Dynamic Boundaries" (SRD) to mark authenticated windows in the browser. [17] This scheme uses a random number generator to set a bit that determines the frequency of border changes (i.e., the browser border alternates between an inset and outset position). The user's browser will display authenticated webpages using a border that "blinks" at the correct frequency. The browser will also display the correct blinking pattern in a reference window. Any windows that blink at a different frequency than the reference window can not be trusted.

A strength of this solution is that rogue servers can not predict the random number chosen by the browser, and therefore it is difficult to simulate borders that blink at the correct frequency. Another advantage is that the user simply needs to compare a window border to the reference window in order to verify it. Weaknesses of this approach are that dynamically blinking borders may be annoying and distracting. The security depends on how many border frequency options are available and how many users can correctly distinguish.

The source code for this proposal is publicly available, and the only tool required by a user is a browser that is modified to support SRD.

YURL. In the YURL proposal, the user's browser maintains a mapping of a public key hash to a "petname", or nickname. [18] When a user visits a page identified by a YURL, the browser prominently displays the petname that the user previously associated with the website. An untrusted site can be recognized by the absence of a corresponding petname.

One advantage of this scheme is that the secret (the petname) is shared with the user himself, rather than with the trusted server. Therefore, an attacker must

be physically present or must compromise the security of the user's browser or computer to obtain the petname. However, because the secret does not change, it can be replayed if it is captured or observed.

The main disadvantage of the scheme is that it requires user customization for each website (i.e., the user must choose and enter a petname). This scheme relies on user memory to recognize the secret phrase and to associate it with the correct website. Therefore, we expect that that the choice of petname will be predictable. For example, a large subset of users will choose "Amazon" for Amazon.com.

We recommend that designers should not rely on the absence of a pet name to identify untrusted windows. Untrusted windows should be marked in a highly visible way that indicates that no petname is present. Otherwise, attackers can insert an image of the petname display into an untrusted page to fool users (especially those users that choose predictable petnames).

The details of the YURL proposal are publicly available, and the only tool required by the user is a browser modified to display petnames.

2.4 Anti-phishing Tools

eBay Toolbar. The eBay Toolbar is a browser plug-in that eBay offers to its customers, primarily to help them keep track of auction sites. [19] The toolbar has a feature, called "AccountGuard" that monitors the domain names that users visit and provides a warning in the form of a colored tab on the toolbar. The tab is usually grey, but it turns green if the user is on an eBay or PayPal site. It turns red if the user is on a site that is known to be a spoof by eBay. The toolbar also allows users to submit suspected spoof sites to eBay.

Because the colored tab is usually grey, users may ignore this feature in the toolbar. However, once they know about the feature, it is easy to recognize a red-colored tab. This scheme requires no effort on the part of the user, other then to notice color changes in the toolbar.

A drawback to this approach is that it only flags sites that are known to be spoofs by eBay and PayPal. Users are unlikely to install several toolbars that apply to other types of websites (though it may be possible to develop one toolbar that would work for a wider range of sites). The main weakness is that there is always a period of time between the time that a spoof is detected and confirmed and when the toolbar can begin reporting spoofs to users. This will allow some users to view a spoofed page without a warning. If spoof reports are not carefully confirmed, denial of service attacks are possible.

eBay has not revealed the security policies or technical details of the toolbar. The only tool required by the user in this scheme is the browser plug-in supplied by eBay.

SpoofGuard. SpoofGuard is an Internet Explorer browser plug-in that examines web pages and warns users when webpages have a high probability of being spoofs [20]. This calculation is performed by examining the URL, images, and links,

comparing them to the stored history and by looking for other characteristics of spoofed sites.

The main weakness of this approach is that the SpoofGuard checks can be evaded by simple modifications to spoof pages. If adopted, SpoofGuard will force phishers to work harder. However, new detection tests must be continuously deployed as phishers become more sophisticated. SpoofGuard makes use of PwdHash [21], an Internet Explorer plug-in that replaces a users password with a one way hash of the password and domain name. As a result, the website only sees the domain specific hash of the password instead of the password itself. This is a simple but useful technique in preventing phishers from collecting user passwords. Unfortunately, it can also create problems where users have multiple accounts at websites with the same domain name, or where account logins occur on pages with different domain names.

This scheme requires no effort on the part of the user other than to notice a status indicator (a red, yellow or green light on the toolbar) and view warning windows (which can optionally be turned off). No additional tools other than a browser plug-in are required.

Spoofstick. Spoofstick is a simple toolbar extension for the Internet Explorer and Mozilla Firefox browsers. [22] The toolbar provides basic information about the domain name of the website. For example, if the user is visiting Ebay, the toolbar displays "You're on ebay.com". If the user is at a spoofed site, the toolbar might instead display "You're on 10.19.32.4". This toolbar can help the user to detect phishing attacks where phishers choose domain names that are syntactically or semantically similar to a legitimate domain name.

Spoofstick is vulnerable to a phishing attack where different websites are opened in multiple frames in the browser window. Spoofstick may display the correct domain name from the legitimate site in one frame, while the user is viewing a site hosted by a rogue server in another frame.

This scheme requires only requires that the user check the domain name listed in the toolbar. It allows the user to customize the appearance of the toolbar, which prevents the toolbar itself from being spoofed (e.g. from being displayed as a static image within the content of an untrusted page).

The technical details of the operation of the toolbar are not publicly available. The user requires no additional tools other than the browser plugin.

3 Dynamic Security Skins

We evaluate a system that we recently proposed, Dynamic Security Skins (DSS). DSS allows a remote server to prove its identity in a way that is easy for a human user to verify and hard for an attacker to spoof[4]. We are implementing DSS as an extension for the Mozilla Firefox browser. Here we briefly review the system with respect to the HIP criteria.

Our extension provides the user with a *trusted password window*. This is a dedicated window for the user to enter usernames and passwords and for

the browser to display security information. We establish a trusted path to the password window by assigning each user a random photographic image that will always appear in that window. The user should easily be able to recognize the personal photo and should only enter his password when this image is displayed. As shown in Figure 1, the photographic image serves as the background of the window, and it is also transparently overlaid onto the textboxes. This ensures that user focus is on the image at the point of text entry and makes it more difficult to spoof the password entry boxes (e.g., by using a pop-up window over that area).

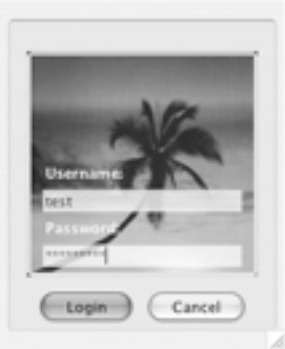

Fig. 1. The trusted password window uses a background image to prevent spoofing of the window and textboxes

We adapt an existing protocol, the Secure Remote Password protocol (SRP) developed by Tom Wu [23], that allows a user and server to authenticate each other over an untrusted network. We chose SRP because it allows us to preserve the familiar use of passwords, without requiring the user to send his password to the server. Instead, the user chooses a password and then applies a one-way function to that secret to generate a *verifier*. The user sends the verifier to the server one time. After the first exchange, the user and the server engage in a series of steps that prove to each other that they hold the verifier, without revealing it. The protocol resists dictionary attacks on the verifier from both passive and active attackers, which allows weak passwords to be used safely.

Assuming that a successful authentication has taken place, how can a user distinguish authenticated web pages from those that are not "secure"? To accomplish this, we adapted the SRP protocol (use of this specific protocol is not a requirement for our approach). In the last step of the protocol, the server presents a proof to the user in the form of a hash value. In our system, the server uses the hash value to generate an abstract image, or *visual hash*. [24] As a visual hash algorithm, we use Random Art [13], which has previously been proposed for use in graphical password user authentication. [24, 25, 26] The remote server can use the image to create a "skin" that modifies the appearance

Fig. 2. The trusted password window displays a visual hash that should match the website image

Fig. 3. The remote website displays a visual hash in the background of a form

of the webpage or particular elements in a web page (e.g., an image can be embedded in a web form that requests sensitive information). The user's browser can independently compute the same image because it also knows the values of the verifier and random parameters that were exchanged by each party during the protocol. The browser presents the user with the image that it expects to receive from the server in the trusted window. If the website image matches the image displayed in the user's trusted window, the user can easily verify that the information request originates from a known party. DSS meets the criteria for a HIP that allows humans to detect phishing attacks. In our system, the user issues a challenge to the server that is:

1. Is easy for a legitimate computer to pass. A legitimate server that holds the user's verifier can display the correct visual hash on its website.
2. Is hard for other computers to pass. A rogue server can not produce a visual hash that will match the users reference image without the user verifier. If the visual hash image is observed or captured, it can not be replayed in subsequent transactions. If users can recognize his personal image, the trusted window is hard to spoof.
3. Produces results that are easy for a human to verify. Creating a trusted path to the password window requires no effort (or a one-time customization for users who wish to change personal images). The user must only recognize one image to verify the trusted window and visually match two images to establish the identity of the server.
4. Uses a protocol that is publicly available. The security of DSS does not rely on the secrecy of the browser or source code or browser extension source code, and the SRP protocol is publicly available.
5. Does not require the user to have specialized tools. DSS requires the user to have a browser that includes our extension. It does not require the user to store or manage any keys. The only secret that must be available to the browser is the user password (that can be a weak password memorized by the user).

4 Conclusions and Future Work

In this paper, we propose a new class of HIPs that allow a human to distinguish one computer from another. In this type of HIP, the user presents a challenge that must be easy for a particular computer to pass, yet hard for other computers to pass. Additionally, the results must be verifiable by a human, and the protocol must be publicly available. This class of HIPs can be used by a human to distinguish a known and legitimate website from an unknown one in order to detect phishing attacks.

We define five properties of an ideal HIP to detect phishing attacks. Using these properties, we evaluate existing and proposed anti-phishing schemes to discover their benefits and weaknesses.

We evaluate a recently proposed system that allows a remote server to prove its identity in a way that is easy for a human user to verify and hard for an attacker to spoof. In our scheme, the server presents its proof in the form of an image that is unique for each user and each transaction. The user's browser can independently compute the image that it expects to receive from the server. To authenticate the server, the user can visually verify that the images match.

We will continue development of the prototype and will conduct a user study to evaluate the effectiveness of image comparison as a technique for users to identify remote servers. In the user study, we will test whether users can reliably recognize their trusted window and whether they can be fooled by spoofed trusted windows and spoofed server images.

References

[1] First Workshop on Human Interactive Proofs. http://www2.parc.com/istl/groups/did/HIP2002/ (2002)

[2] Litan, A.: Phishing Attack Victims Likely Targets for Identity Theft. Gartner Research FT-22-8873 (2004)

[3] Loftesness, S.: Responding to Phishing Attacks. http://www.glenbrook.com/opinions/phishing.htm (2004)

[4] Dhamija, R., Tygar, J.D.: (Phishing: A Model Problem for Usability in Privacy and Security (To Appear))

[5] Netcraft: SSLs Credibility as Phishing Defense is Tested. http://news.netcraft.com/archives/2004/03/08/ssls_credibility_as_phishing_defense_is_tested.html (2004)

[6] Microsoft: Erroneous Verisign Issued Digital Certificates Pose Spoofing Hazard. Technical Report Microsoft Security Bulletin MS01-017 (2001)

[7] Herzberg, A., Gbara, A.: Protecting (even) Naive Web Users, or: Preventing Spoofing and Establishing Credentials of Websites. Technical Report Draft of July 2004 (2004)

[8] Pretty Good Privacy. (www.pgp.com/)

[9] Verisign: Verisign Secured Seal Program. (http://www.verisign.com/products-services/security-services/secured-seal/)

[10] TRUSTe. (http://www.truste.org/)

[11] RSA Security: America Online and RSA Security Launch AOL PassCode Premium Service. http://www.rsasecurity.com/press_release.asp?doc_id=5033 (2004)

[12] RSA Security: Protecting Against Phishing by Implementing Strong Two-Factor Authentication. https://www.rsasecurity.com/products/securid/whitepapers/PHISH_WP_0904.pdf (2004)

[13] Pullar-Strecker, T.: NZ bank adds security online. (The Sydney Morning Herald, November 8, 2004)

[14] Passmark Security: Protecting Your Customers from Phishing Attacks: an Introduction to Passmarks. (http://www.passmarksecurity.com/)

[15] Visa USA: Verified by Visa. (https://usa.visa.com/personal/security/vbv/)

[16] Haber, R.N.: How We Remember What We See. Scientific American **222** (1970) 104–112

[17] Zishuang, Y., Smith, S.: Trusted Paths for Browsers. In: Proceedings of the 11th USENIX Security Symposium,, IEEE Computer Society Press (2002)

[18] Waterken Inc.: Waterken YURL Trust Management for Humans. http://www.waterken.com/dev/YURL/Name/ (2004)

[19] eBay: eBay Toolbar. (http://pages.ebay.com/ebay_toolbar/)

[20] Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., Mitchell, J.C.: Client Side Defense Against Web-based Identity Theft. (http://crypto.stanford.edu/SpoofGuard/)

[21] Ross, B., Jackson, C., Miyake, N., Boneh, D., Mitchell, J.C.: A Browser Plug-in Solution to the Unique Password Problem. Technical Report Stanford-SecLab-TR-2005-1 (2005)

[22] Core Street: SpoofStick. (www.corestreet.com/spoofstick/)

[23] Wu, T.: The Secure Remote Password Protocol. In: Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium, San Diego, CA. (1998)

[24] Perrig, A., Song, D.: Hash Visualization: A New Technique to Improve Real World Security. In: International Workshop on Cryptographic Techniques and E-Commerce. (1999)
[25] Dhamija, R.: Hash Visualization in User Authentication. In: Proceedings of the Computer Human Interaction Conference Short Papers. (2000)
[26] Dhamija, R., Perrig, A.: Déjà Vu: A User Study. Using Images for Authentication. In: Proceedings of the 9th USENIX Security Symposium. (2000)