Fighting Phishing at the User Interface

by

Min Wu

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

[September 2006]
August 2006

Author ....
Department of Electrical Engineering and Computer Science
August 21, 2006

Certified by ....
Robert C. Miller
Assistant Professor
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Fighting Phishing at the User Interface

by

Min Wu

Submitted to the Department of Electrical Engineering and Computer Science on August 21, 2006, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering

Abstract

The problem that this thesis concentrates on is phishing attacks. Phishing attacks use email messages and web sites designed to look as if they come from a known and legitimate organization, in order to deceive users into submitting their personal, financial, or computer account information online at those fake web sites.

Phishing is a semantic attack. The fundamental problem of phishing is that when a user submits sensitive information online under an attack, his mental model about this submission is different from the system model that actually performs this submission. Specifically, the system sends the data to a different web site from the one where the user intends to submit the data. The fundamental solution to phishing is to bridge the semantic gap between the user's mental model and the system model.

The user interface is where human users interact with the computer system. It is where a user's intention transforms into a system operation. It is where the semantic gap happens under phishing attacks. And therefore, it is where the phishing should be solved.

There are two major approaches to bridge the semantic gap at the user interface. One approach is to reflect the system model to the user. Anti-phishing toolbars and the browser's security indicators take this approach. User studies in this thesis show that this approach is not effective at preventing phishing. Users are required to constantly pay attention to the toolbar and are expected to have the expertise to always correctly interpret the toolbar message. Normal users meet neither of these requirements.

The other approach is to let users tell the system their intentions when they are submitting data online. The system can then check if the actual submission meets the user's intention or not. If there is a semantic gap, the system can effectively warn the user about this discrepancy and provide a safe path to the user's intended site. Web Wallet, designed and implemented as a new anti-phishing solution, takes this approach. It is a dedicated browser sidebar for users to submit their sensitive information online. User studies in this thesis shows that Web Wallet is not only an effective and promising anti-phishing solution but also a usable personal information manager.

Thesis Supervisor: Robert C. Miller
Title: Assistant Professor

Acknowledgments

At first, I would like to thank my thesis advisor, Professor Robert Miller. This thesis cannot be done without his fully support. I have worked with him for years on this thesis. All the research behind this thesis involves his time, his resources, his ideas, his judgments, and his encouragements. He not only acutely pinpointed even the tiny problems existing in my research but also encouraged me to stand strongly on the valuable parts of it. He taught me how to design and implement a valid user study and to analyze the results. He taught me that I should not only concentrate on the research ideas but also pay attention to coding, writing and presentation skills. He encouraged me not only to be focus on my own research but also to get involved in the whole academia. He shows me in person what a scientist should be. I am extremely fortunate to have him as my research advisor. The things that I have learned from him during the past four years will benefit my whole life.

I would like to thank my thesis readers, Dr. Ruth Rosenholtz and Professor Srini Devadas. Both of them have offered numerous comments and valuable feedbacks about my thesis. Dr. Rosenholtz taught me how to do a valid statistical analysis and the formal way to present the analysis result in papers and presentations. She also provided me an opportunity to learn and use the eye tracker as a monitoring tool during a user study.

I would like to thank Simson Garfinkel. I have worked with him on usability and security for three years. I have got numerous feedbacks and suggestions from him to my research. The cell-phone authentication project presented in chapter 4 was originally from his idea. To me, he is a computer scientist who knows every thing about practical computer security. He taught me that I should not only concentrate on the pure research but also think about how to make the research project to be benefit to the society as soon as possible.

I would like to thank two persons from the industry. Steve Strassman from Orange originally suggested the "choose and approve" interface in the cell-phone authentication project presented in chapter 4. Based on his suggestion, I eventually propose one of the user interface design principles about integrating a security decision into user's workflow.

Mike Rowan from GeoTrust gave me tremendous help in the Web Wallet implementation. He allowed me freely access the TrustWatch service of GeoTrust in order to get the site reputation information. This service has saved me a lot of time and effort in building Web Wallet as a full-featured application. He explained to me in detail the backend implementation how TrustWatch decides a site reputation. He also let me access the valuable phishing report service of GeoTrust so that I can collect fresh phishing sites before they have been blocked or taken down in order to be updated with the state-of-the-art phishing attacks.

I would like to thank the User Interface Design group at MIT CSAIL. The good atmosphere and constant support from this group help my research a lot. All the members have been my pilot users without any payment in order to test bugs and problems of a user study procedure before I recruited real users.

I would like to thank the members of the CHISEC reading group at MIT, especially Richard Conlan. We met once per two weeks to discuss papers about usability and security. I get a lot of valuable ideas from this group meeting.

Finally, I need to express my thanks, appreciation and gratitude to my great wife Yan Zhang and my adorable daughter Julia. Yan has constantly given me full support to my research by taking care of everything of the family. She has also made incredible sacrifice to make this thesis come true.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is phishing?

As people increasingly rely on the Internet for business, personal finance and investment, Internet fraud becomes a greater and greater threat. Internet fraud takes many forms, from phony items offered for sale on eBay, to scurrilous rumors that manipulate stock prices, to scams that promise great riches if the victim will help a foreign financial transaction through his own bank account.

One interesting species of Internet fraud is phishing. Phishing attacks use email messages and web sites designed to look as if they come from a known and legitimate organization, in order to deceive users into disclosing personal, financial, or computer account information. The attacker can then use this information for criminal purposes, such as identity theft, larceny, or fraud. Users are tricked into disclosing their information either by providing it through a web form or by downloading and installing hostile software.

Phishing has been growing really fast, as shown in Figure 1-1. According to the Anti-Phishing Working Group, in the past two years, the number of unique reported phishing attacks per month has increased more than 160 times and the number of unique reported phishing sites per month has increased about 16 times. In June 2006, 130 legitimate brands have been attacked. [32] Financial services were the most targeted industry sector (figure 1-2).

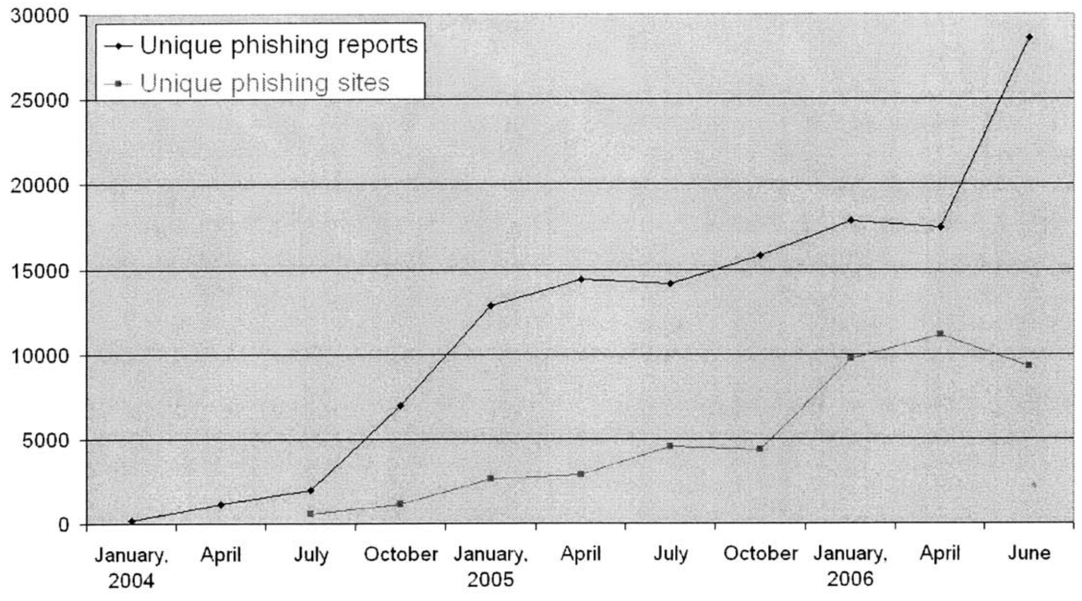


Figure 1-1: Numbers of unique reported phishing attacks and phishing sites per month from 2004 to 2006 (source: Anti-Phishing Working Group)

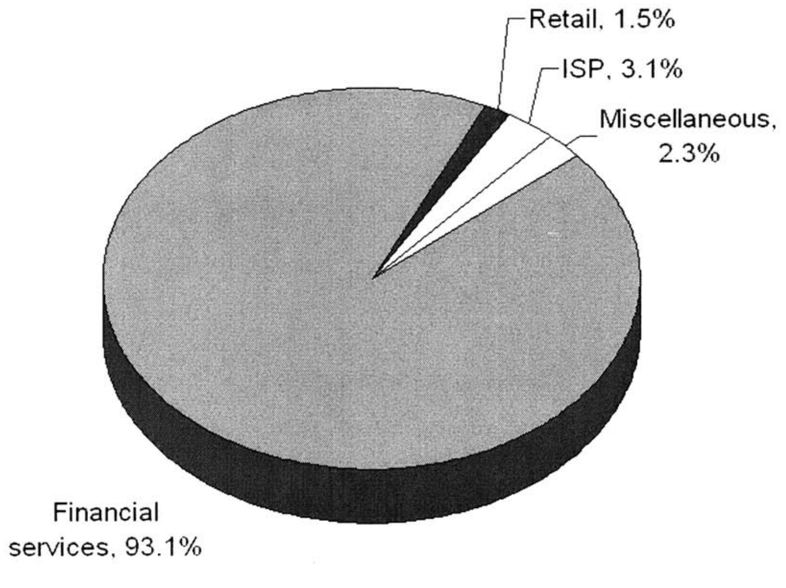


Figure 1-2: Targeted industry sectors by phishing in June 2006 (source: Anti-Phishing Working Group)

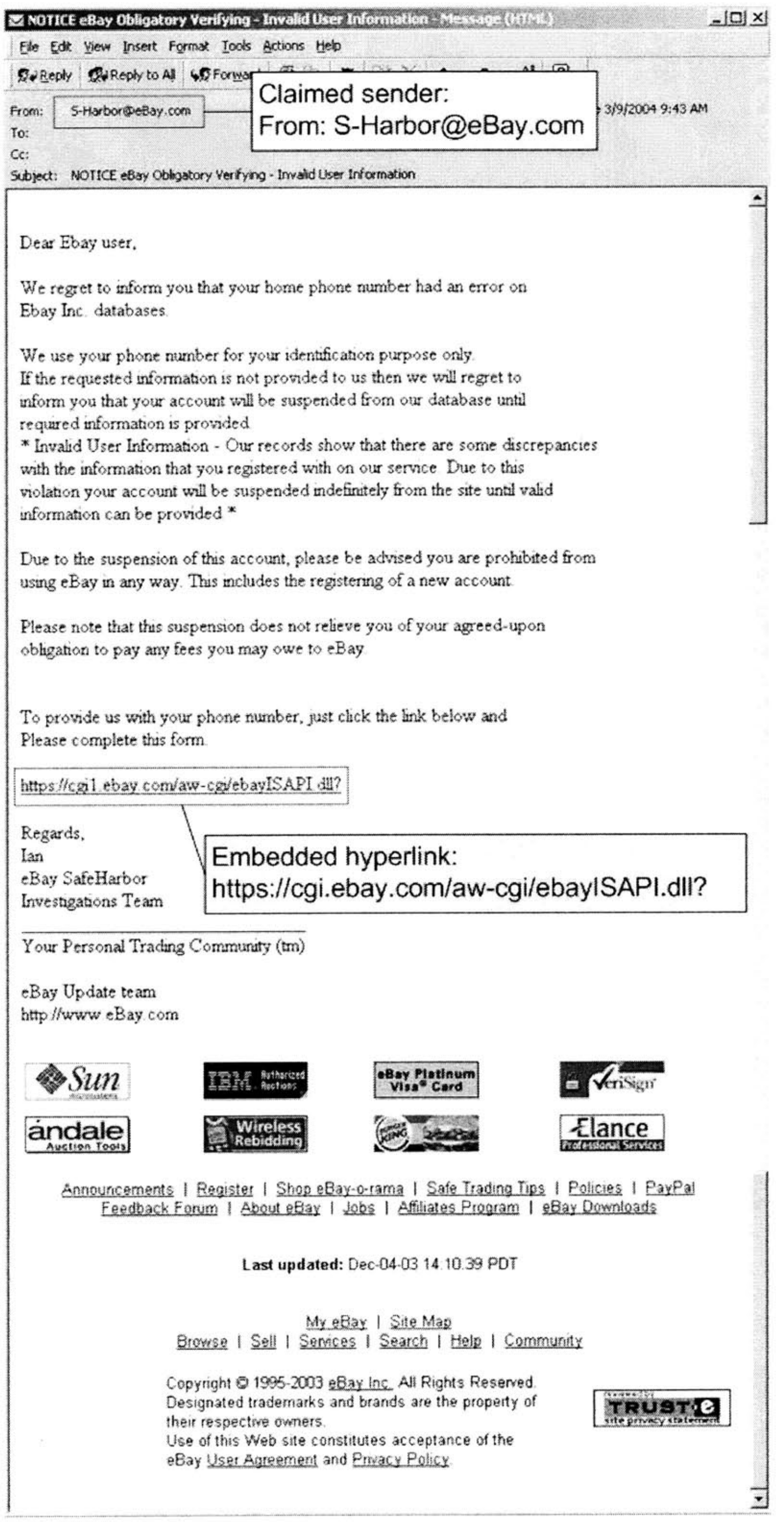
Phishing is costly. It causes big financial losses for organizations that support online financial activities. The Gartner group estimated that the direct phishing-related loss to US banks and credit card issuers in 2003 was $1.2 billion. [20] A report in 2004 estimated that phishing costs financial institutions about $100,000 to $150,000 per attack, mainly because phishing attacks compromise the brand, the most important asset, of the targeted financial institutes. "There's a lot of brand risk. It's very different when one million people are getting e-mails from you. Are they likely to continue to do business with you?". [43]

Phishing attacks have a big negative impact on consumer confidence about e-commerce. Gartner's 2005 survey estimated that phishing attacks will inhibit three-year U.S. e-commerce growth rates by 1% to 3%. The survey also showed that 80% of users stated that online attacks have impacted their trust in email from companies or individuals they do not know personally; 33% stated that they would buy fewer items online than they otherwise would; 14% stated that they would stop paying bills via online banking; and 4% stated that they would stop online banking altogether. [46, 66]

1.2 Anatomy of a real phishing attack

The Anti-Phishing Working Group (APWG) collects and archives examples of phishing attacks. It is a valuable service because phishing web sites exist only for a short time before they get shut down or blocked. One example on APWG is an attack against eBay customers, first reported on March 9, 2004 [31]. The attack begins when a potential victim receives an email (figure 1-3), purporting to be from eBay, that claims that the user's account information is invalid and must be corrected. The email contains an embedded hyperlink that appears to point to a page on eBay's web site. This web page (figure 1-4) asks for the user's credit card number, contact information, Social Security Number, and eBay username and password.

Beneath the surface, however, neither the email message nor the web page is what it appears to be. Figure 1-5 breaks the deception down schematically. The phishing email resembles a legitimate email from eBay. Its source (listed in the "From:"


NOTICE eBay Obligatory Verifying - Invalid User Information - Message (HTML)
File Edit View Insert Format Tools Actions Help
Reply
Reply to All
Forward

Claimed sender:
From: S-Harbor@eBay.com

From: S-Harbor@eBay.com 3/9/2004 9:43 AM
To:
Cc:
Subject: NOTICE eBay Obligatory Verifying - Invalid User Information

Dear Ebay user,

We regret to inform you that your home phone number had an error on
Ebay Inc. databases.

We use your phone number for your identification purpose only.
If the requested information is not provided to us then we will regret to
inform you that your account will be suspended from our database until
required information is provided.
* Invalid User Information - Our records show that there are some discrepancies
with the information that you registered with on our service. Due to this
violation your account will be suspended indefinitely from the site until valid
information can be provided *

Due to the suspension of this account, please be advised you are prohibited from
using eBay in any way. This includes the registering of a new account.

Please note that this suspension does not relieve you of your agreed-upon
obligation to pay any fees you may owe to eBay.

To provide us with your phone number, just click the link below and
Please complete this form.

https://cgi1.ebay.com/aw-cgi/ebayISAPI.dll?

Embedded hyperlink:
https://cgi.ebay.com/aw-cgi/ebayISAPI.dll?

Regards,
Ian
eBay SafeHarbor
Investigations Team

Your Personal Trading Community (tm)

eBay Update team
http://www eBay com

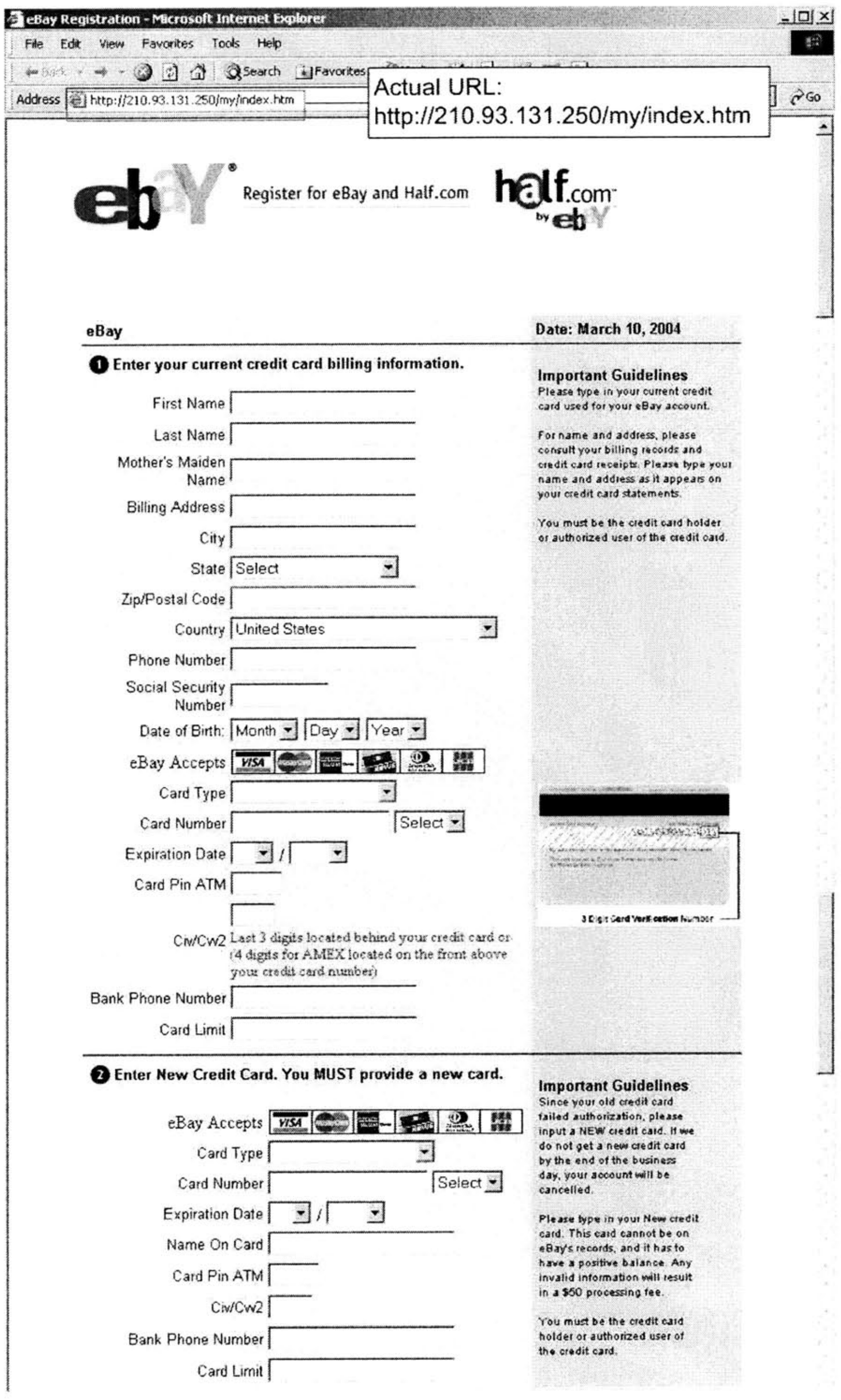
Announcements | Register | Shop eBay-o-rama | Safe Trading Tips | Policies | PayPal
Feedback Forum | About eBay | Jobs | Affiliates Program | eBay Downloads

Last updated: Dec-04-03 14:10:39 PDT

My eBay | Site Map
Browse | Sell | Services | Search | Help | Community

Copyright © 1995-2003 eBay Inc. All Rights Reserved.
Designated trademarks and brands are the property of
their respective owners.
Use of this Web site constitutes acceptance of the
eBay User Agreement and Privacy Policy.

Figure 1-3: Screenshot of the eBay phishing email (source: Anti-Phishing Working Group)


eBay Registration - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Search Favorites
Address http://210.93.131.250/my/index.htm
Go

Actual URL:
http://210.93.131.250/my/index.htm

Register for eBay and Half.com

eBay **Date: March 10, 2004**

1 Enter your current credit card billing information.

First Name
Last Name
Mother's Maiden Name
Billing Address
City
State Select
Zip/Postal Code
Country United States
Phone Number
Social Security Number
Date of Birth: Month Day Year
eBay Accepts
Card Type
Card Number Select
Expiration Date /
Card Pin ATM
Cvv/Cvv2 Last 3 digits located behind your credit card or (4 digits for AMEX located on the front above your credit card number)
Bank Phone Number
Card Limit

Important Guidelines
Please type in your current credit card used for your eBay account.

For name and address, please consult your billing records and credit card receipts. Please type your name and address as it appears on your credit card statements.

You must be the credit card holder or authorized user of the credit card.

3 Digit Card Verification Number

2 Enter New Credit Card. You MUST provide a new card.

eBay Accepts
Card Type
Card Number Select
Expiration Date /
Name On Card
Card Pin ATM
Cvv/Cvv2
Bank Phone Number
Card Limit

Important Guidelines
Since your old credit card failed authorization, please input a NEW credit card. If we do not get a new credit card by the end of the business day, your account will be cancelled.

Please type in your New credit card. This card cannot be on eBay's records, and it has to have a positive balance. Any invalid information will result in a $50 processing fee.

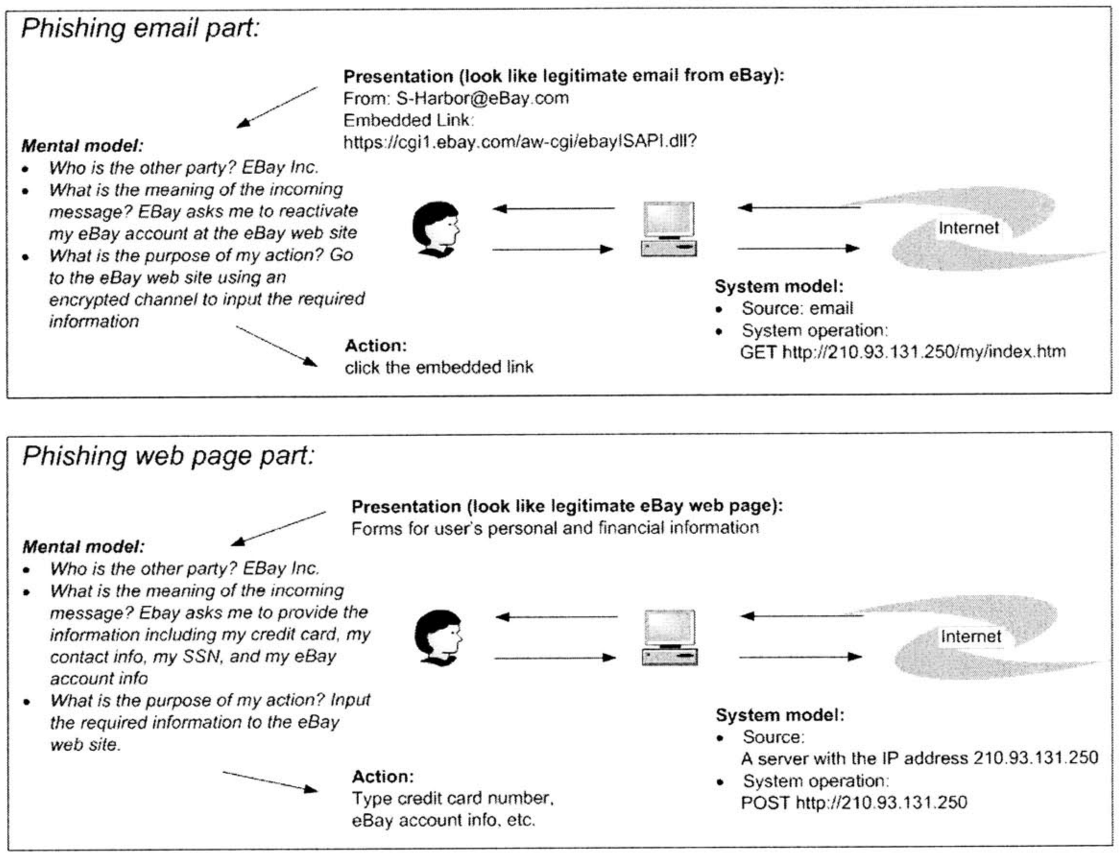
You must be the credit card holder or authorized user of the credit card.

Figure 1-4: Screenshot of the eBay phishing web page (source: Anti-Phishing Working Group)

header) appears to be S-Harbor@eBay.com, which refers to the legitimate domain name for eBay Inc. The link embedded in the message also appears to go to eBay.com, even using an encrypted channel ("https:"). Based on these presentation cues and the content of the message, the user forms a mental model of the message: eBay is requesting updated information. The user then performs an action, clicking on the embedded hyperlink, which is presumed to go to eBay. But the user's action is translated into a completely different system operation – namely, retrieving a web page from IP address 210.93.131.250, a server from a communication company registered in Seoul, South Korea. This company has no relationship with eBay Inc.

The phishing web site follows a similar pattern of deception. The page looks like a legitimate eBay web page. It contains an eBay logo, and its content and layout match the format of pages from the actual eBay web site. Based on this presentation, the user forms a mental model that the browser is showing the eBay web site and that the requested information must be provided in order to keep the user's eBay account active. The user then performs an action, typing in personal and financial data and clicking the Submit button, with the intention of sending this information to eBay. This action is translated by the web browser into a system operation, encoding the entered data into an HTTP request sent to 210.93.131.250, which is not a legitimate eBay server.

The specific phishing attack addressed in this thesis is *deceptive phishing*, which tricks users into voluntarily leaking their personal information online, just as the above example shows. This is a pure semantic attack, without taking any advantage of the system vulnerabilities. Another kind of phishing attack is *malware-based phishing*, which tricks users into modifying their computer system by either directing users to a phishing site and stealthily installing a keylogger or screenlogger, or asking them to download and install a virus that claims to be a security patch [41], or urging them to delete some useful files that are claimed to be a dangerous virus [21]. The design principles that will be proposed in chapter 5 to solve deceptive phishing can be applied to solve malware-based phishing too, as will be discussed in section 9.2.2.



Phishing email part:

Presentation (look like legitimate email from eBay):
From: S-Harbor@eBay.com
Embedded Link:
https://cgi1.ebay.com/aw-cgi/ebayISAPI.dll?

Mental model:

- *Who is the other party? EBay Inc.*
- *What is the meaning of the incoming message? EBay asks me to reactivate my eBay account at the eBay web site*
- *What is the purpose of my action? Go to the eBay web site using an encrypted channel to input the required information*

Action:
click the embedded link

System model:

- Source: email
- System operation:
  GET http://210.93.131.250/my/index.htm

Phishing web page part:

Presentation (look like legitimate eBay web page):
Forms for user's personal and financial information

Mental model:

- *Who is the other party? EBay Inc.*
- *What is the meaning of the incoming message? Ebay asks me to provide the information including my credit card, my contact info, my SSN, and my eBay account info*
- *What is the purpose of my action? Input the required information to the eBay web site.*

Action:
Type credit card number,
eBay account info, etc.

System model:

- Source:
  A server with the IP address 210.93.131.250
- System operation:
  POST http://210.93.131.250

Figure 1-5: Anatomy of the eBay phishing attack

1.3 Phishing as a semantic attack

Bruce Schneier has observed that methods for attacking computer networks can be categorized in waves of increasing sophistication and abstraction. According to Schneier, the first wave of attacks was physical in nature, targeting the computers, the network devices, and the wires between them, in order to disrupt the flow of information. The second wave consisted of syntactic attacks, which target vulnerabilities in network protocols, encryption algorithms, or software implementations. Syntactic attacks have been a primary concern of security research for the last decade. The third wave is semantic: "attacks that target the way we, as humans, assign meaning to content." [59]

Phishing is a semantic attack. Making the computer system robust is not sufficient to solve phishing because phishing does not exploit system bugs. The system does what it is exactly supposed to do. Moreover, phishing tricks users into interacting with the attacker's message through their computer, which is different from other types of semantic attacks like misinformation, scams, and hoaxes. Successful phishing depends on a discrepancy between the way a user perceives a communication, like an email message or a web page, and the actual effect of the communication. Figure 1-6 shows the structure of a typical Internet communication, dividing it into two parts. The system model is concerned with how computers exchange bits – protocols, representations, and software. When human users play a role in the communication, however, protecting the system model is not enough, because the real message communicated depends not on the bits exchanged but on the semantic meanings that are derived from the bits. This semantic layer is the user's mental model. The effectiveness of phishing indicates that human users do not always assign the proper semantic meaning to their online interactions.

When a user faces a phishing attack, the user's mental model about the interaction disagrees with the system model. For example, the user's intention may be "go to eBay," but the actual implementation of the hyperlink may be "go to a server in South Korea." It is this discrepancy that enables the attack, and it is this discrepancy

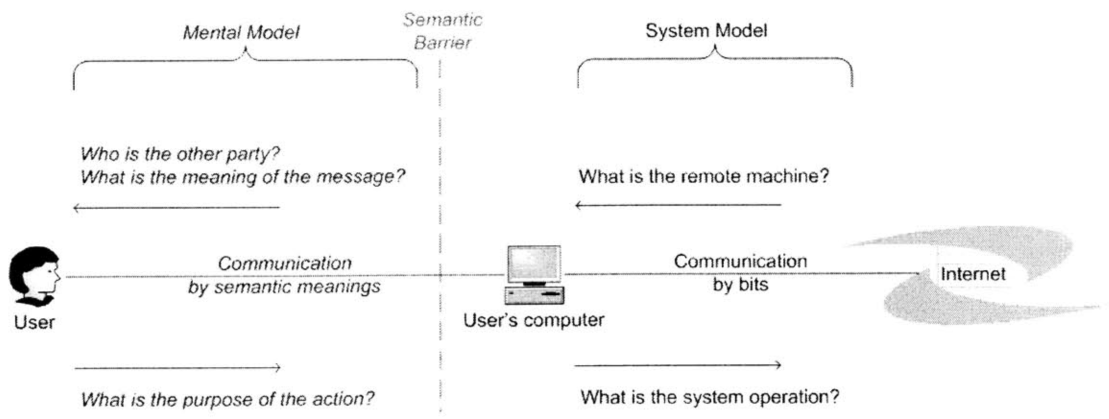

Mental Model
Semantic Barrier
System Model
Who is the other party?
What is the meaning of the message?
What is the remote machine?
Communication by semantic meanings
Communication by bits
Internet
User
User's computer
What is the purpose of the action?
What is the system operation?

Figure 1-6: Human-computer interaction has a semantic barrier between a user and his computer

that makes phishing attacks very hard to defend against. I call this discrepancy a *semantic gap* between the user's mental model and the system model. Users derive their mental models of the interaction from the presentation of the interaction – the way it appears on the screen. The implementation details of web pages and email messages are hidden, and are generally inaccessible to most users. Thus, the user is in no position to compare his mental model with the system model, and it would take extra effort to do so. On the other hand, email clients and web browsers follow the coded instructions provided to them in the message, but are unable to check the user's intentions. Without awareness of both models, neither the user nor the computer is able to detect the semantic gap introduced by phishing.

One extreme solution to the phishing problem would simply discard the presentation part of an Internet communication – the part that produces the user's mental model – because it cannot be trusted. Instead, a new presentation would be generated directly from the implementation. If the user's computer is trustworthy, then, the presentation seen by the user would be guaranteed to be related to the actual implementation. Unfortunately, the cost of this idea in both usability and functionality would be enormous. Most online messages are legitimate, after all, with the presentation correctly reflecting the implementation. Phishing messages are rare (but pernicious) exceptions. So this solution would improperly sacrifice the freedom of legitimate senders to present and brand themselves in order to block a small number of wrongdoers.

So the fact must be accepted that users will see messages with mismatched presentation and implementation. Attempts to fight phishing computationally try to enable the computer to bridge the semantic gap. But the human user must be the final decision-maker about whether a message is phishing. The reason is that phishing targets how users assign semantic meaning to their online interactions, and this assignment process is outside the system's control.

1.4 The user interface is the right place to solve phishing

The user interface is where the human users interact with the computer systems. It is where the user's intention transforms into the system operation. It is where the semantic gap arises. And thus it is where phishing should be solved.

One typical anti-phishing approach is to use visual indicators, like an informative toolbar, to differentiate legitimate messages from phishing messages. This approach tries to bridge the semantic gap by reflecting the system model to human users and expect them to make a wise decision under phishing attacks. User studies in this thesis show that the tested anti-phishing toolbars fail to effectively prevent high-quality phishing attacks. Many subjects failed to constantly pay attention to the toolbar's messages; others disregarded or explained away the toolbars' warnings if the content of web pages looked legitimate. The studies also found that many subjects do not understand phishing attacks or realize how sophisticated such attacks can be.

Two-factor authentication has also been used to prevent phishing attacks. In this thesis, I propose an authentication protocol by using a cell phone as an authentication token for users to log in to web sites. This protocol is proved to be secure and has been tested to be more user-friendly than the existing cell-phone authentication solutions.

Based on the lessons learned from the anti-toolbar studies and the cell-phone authentication project, three user interface design principles are proposed to solve phishing:

- **Intention-centered design.** Instead of showing the system model to a user as the toolbar solution does, it is better to let the user tell the system his mental model, especially his intention for the current operation. The system can then figure out if its operation matches the user's intention. As mentioned before, under phishing attacks, the system does different jobs than what the user wants the system to do. And with the user's intention, the system can effectively warn him about this mismatch, which is the fundamental danger of phishing. Moreover, the system can provide an alternative *safe path* to fulfill the user's intention.

- **Integrate security into the user's workflow.** Security as a separate subtask from the user's main task can be easily ignored. By integrating the security decisions into the user's workflow, a user has to acknowledge and deal with the security mechanism.

- **Make the safest path the path of least resistance.** To prevent a user from bypassing the security mechanism, the security mechanism should be in the most natural and the easiest workflow that the user is going to take to finish their tasks.

Web Wallet is proposed as a new anti-phishing solution incorporating these design principles. It is a dedicated browser sidebar for users to submit their sensitive information online. It automatically detects and disables web forms that request user's sensitive information. In order to submit sensitive information online, a user presses a security key on the keyboard to open Web Wallet, where he types his data or retrieves data that he has previously stored into Web Wallet. The data is then filled into the web forms. Before the fill-in, however, Web Wallet checks if the current site is good enough to receive the submitted sensitive data. If the current site is not qualified as a good site, Web Wallet requires the user to explicitly indicate where he wants his data to go. If the user's intended site is not the current site, which probably indicates phishing, Web Wallet shows a warning about this discrepancy, and gives the user a safe path to his intended site.

A user study of Web Wallet found that it is a promising anti-phishing approach. In the study, Web Wallet significantly decreased the spoof rate of normal phishing attacks from 63% to 7%. It effectively prevents all tested types of phishing attacks as long as it was open and in use. A majority of the subjects successfully learned to depend on Web Wallet to submit their sensitive information.

Web Wallet is implemented as Browser Helper Object for Internet Explorer, the world's most popular browser [60]. It is not only an anti-phishing solution but also a secure and usable personal information manager.

1.5 Thesis statement

Phishing attacks can be effectively prevented at the user interface. By letting users indicate their real intention to the system, the semantic gap between the user's mental model and the system model of the human-Internet interaction can be bridged. By integrating the security decision into the user's workflow, security is not a separate sub-task that can be easily ignored or bypassed by users anymore.

1.6 Thesis contribution

This thesis makes the following contributions:

- A user study scenario and methodology that can be used to test anti-phishing solutions in the lab, along with benchmark tasks.

- Experimental results showing why phishing attacks are effective at tricking users into leaking their personal information and why anti-phishing toolbars and security visual indicators cannot effectively prevent phishing attacks.

- A set of user interface design principles that can be used to effectively prevent phishing attacks.
- A novel interface called Web Wallet that uses the proposed the design principles to prevent phishing attacks.
- A cell-phone-based two-factor authentication solution to prevent phishing attacks and hostile keyloggers.
- A novel confirmation interface to prevent users blindly confirming security messages.
- A novel form detection mechanism that detects web forms asking for sensitive personal information.
- An implementation of Web Wallet as an Internet Explorer plug-in as both an anti-phishing solution and a secure personal information manager for users to securely submit their personal information online.

1.7 Outline of the thesis

Chapter 2 makes a survey of the current anti-phishing solutions and phishing-related user studies. Chapter 3 discusses two user studies of the anti-phishing toolbars. Chapter 4 presents a cell-phone-based authentication solution and proposes a mechanism to make the security visual indicators usable. Chapter 5 introduces Web Wallet user interface and its design principles. Chapter 6 introduces the Web Wallet's sensitive-input detection algorithm. Chapter 7 presents a user study in order to evaluate the effectiveness of Web Wallet to prevent phishing. Chapter 8 presents a usability evaluation of Web Wallet. Chapter 9 concludes this thesis by giving anti-phishing recommendations and discussing the future work of Web Wallet.

Chapter 2

Related Work

2.1 Anti-phishing solutions

An online interaction can be separated into four steps (figure 2-1):

- *Message retrieval*: An email message or web page arrives at a user's personal computer from the Internet.
- *Message presentation*: The message is displayed in the user interface, the user perceives it and forms a mental model.
- *User action*: Guided by the mental model, the user performs an action in the user interface, such as clicking a link or filling in a form.
- *System operation*: The user's action is translated into system operations, such as connecting to a web server and submitting data.

Previous approaches to fighting phishing can be classified by which of these four steps they address.

2.1.1 Message retrieval

Ideally, the best defense against phishing would simply block all phishing messages from being shown to the user, by filtering them at the message retrieval step. The

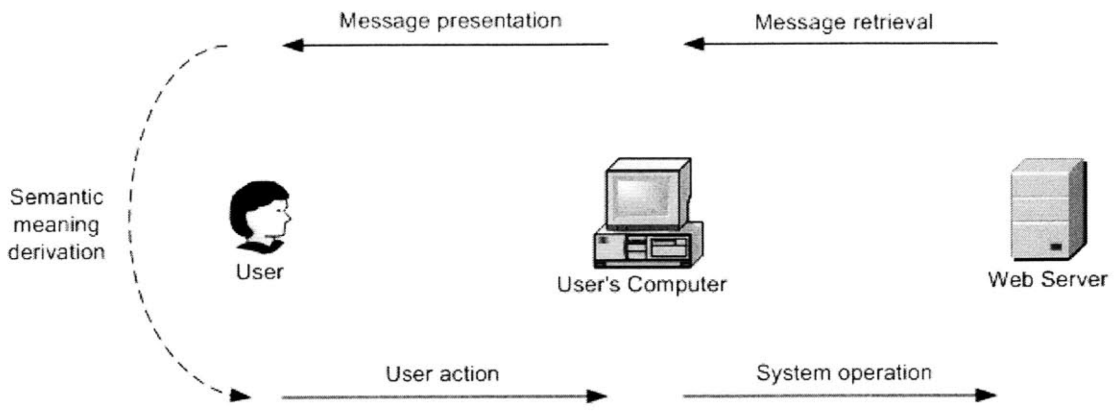


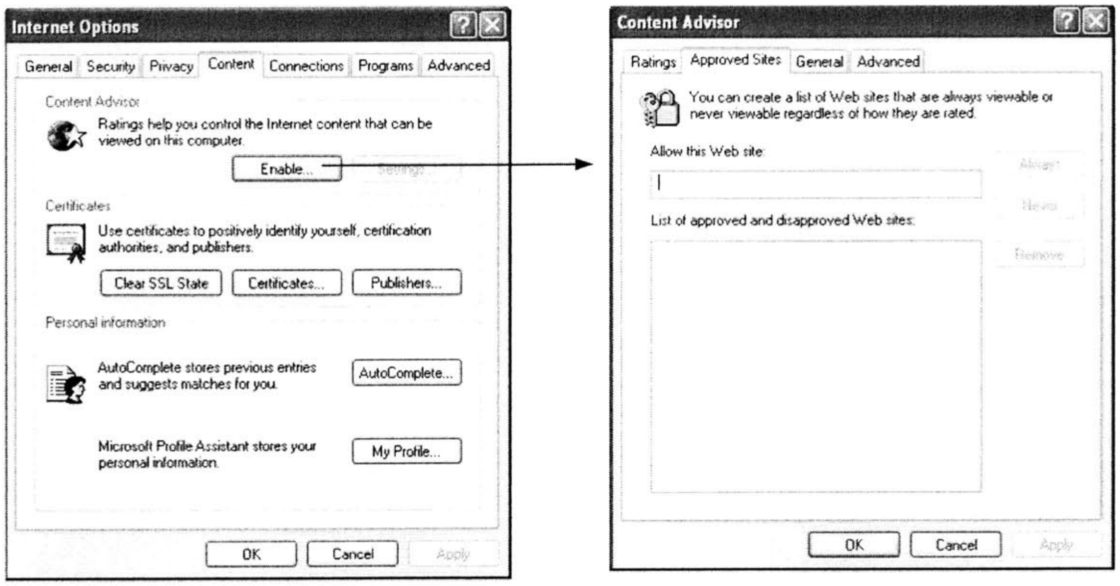
Figure 2-1: Four steps of an online interaction

essential requirement for this solution is that the computer alone must be able to accurately differentiate phishing messages from legitimate ones. Defenses that filter at the message retrieval step depend on message properties that are easily understood by a computer.

Identity of the sender

One of these properties is the identity of the sender. Blacklisting is widely used to block potentially dangerous or unwelcome messages, such as spam. If the sender's IP address is found in a blacklist, the incoming message can be categorized as spam or even simply rejected without informing the user. A blacklist may be managed by an individual user, the approach taken by Internet Explorer's Content Advisor (figure 2-2). Alternatively, it may be managed by an organization or by collaboration among many users.

Blacklisting is widely used for phishing as well. Many existing security toolbars [16, 19, 28, 29, 36, 52, 64] use a blacklist to warn users about phishing sites that have been detected and reported. While most toolbars simply display a warning message, some [16, 29, 64] go further by blocking the entire phishing page (figure 2-3). The security toolbar studies, which will be discussed in Chapter 3, showed that active interruption that blocks the phishing page is far more effective than the passive



Ratings help you control the Internet content that can be viewed on this computer.

Use certificates to positively identify yourself, certification authorities, and publishers.

AutoComplete stores previous entries and suggests matches for you.

Microsoft Profile Assistant stores your personal information.

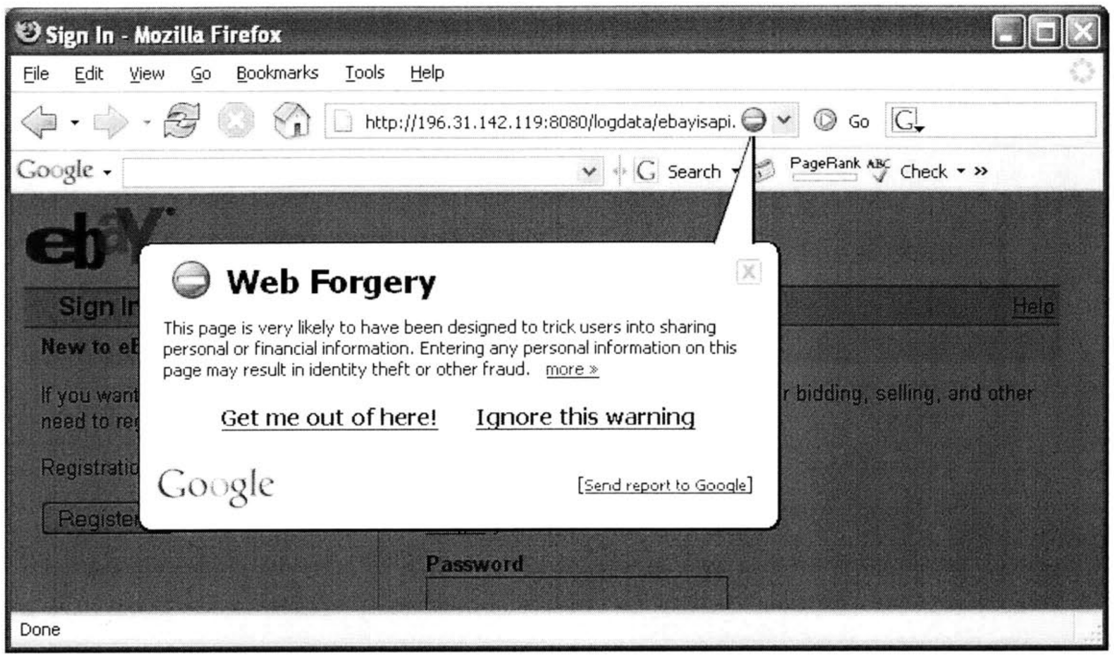
You can create a list of Web sites that are always viewable or never viewable regardless of how they are rated.

Figure 2-2: Internet Explorer's content advisor

warnings displayed in the toolbars. But the biggest problem that prevents blacklisting from being an effective anti-phishing solution is the need to keep it up to date. It is far too easy to generate new identities on the Internet, such as new email addresses and new domain names. New IP addresses are cheap and easy to obtain. Because phishing sites exist for only a short time, a blacklist must be updated within hours or minutes in order to be effective at blocking the attack. Figure 2-4 shows that the two security toolbars with a blacklist failed to detect a phishing site. The site was reported to the Anti-Phishing Working Group on August 06, 2004 and the toolbars were tested against the site on August 12, 2004. Both toolbars' blacklists failed to update with the reported phishing site for six days.

There are two ways to update a blacklist, either periodically synchronizing with a blacklist source or checking every new URL that a user is about to browse at a blacklist source. The latter approach has some privacy concerns, because the blacklist source can then know the user's browsing activity based on the submitted URLs.

Another problem of blacklisting is that it only warns users about the fraudulent site and advises them not to proceed. Users may take the risk of continuing if they


Sign In - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://196.31.142.119:8080/logdata/ebayisapi.
Google Search PageRank Check
Web Forgery
Get me out of here!
Ignore this warning
Google
[Send report to Google]
Password
Done

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. more »

Figure 2-3: Google Safe Browsing for Firefox blackens and blocks the whole page from a detected phishing site

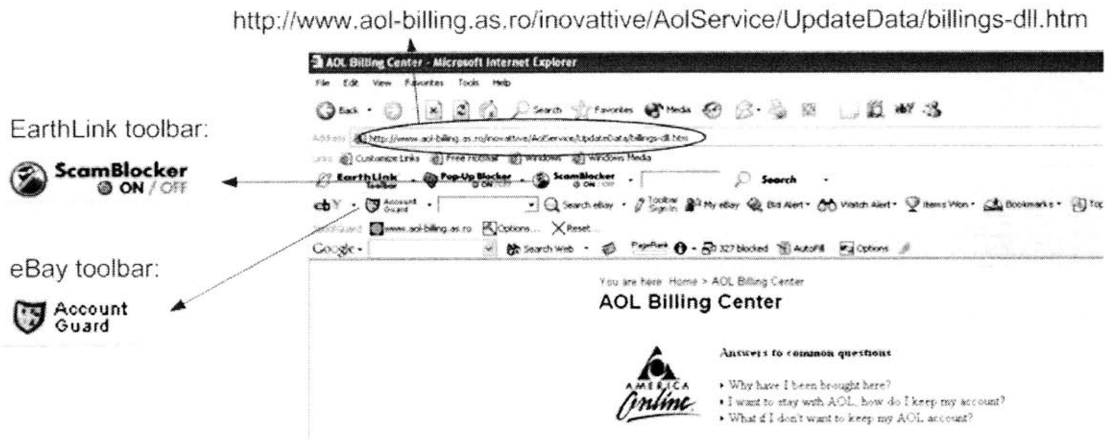

http://www.aol-billing.as.ro/inovattive/AolService/UpdateData/billings-dll.htm
EarthLink toolbar:
ScamBlocker ON / OFF
eBay toolbar:
Account Guard
AOL Billing Center - Microsoft Internet Explorer
AOL Billing Center
Answers to common questions
Why have I been brought here?
I want to stay with AOL, how do I keep my account?
What if I don't want to keep my AOL account?

Figure 2-4: Security toolbars with a blacklist failed to detect a phishing site (tested six days later after the phishing site was reported)

think that the current (fake) task is important to finish. Since most phishing blacklists are maintained by human experts, the legitimate web site being spoofed by each phishing site is known and should be included in the warning message, because it is the site where users intend to go.

The converse of blacklisting is whitelisting, allowing users to see messages only from a list of acceptable sources. For example, Secure Browser [65] controls where users can browse on the Internet using a list of permitted URLs. Whitelisting avoids the new-identity problem because newly created sources are initially marked as unacceptable. But defining a whitelist is a serious problem. Because it is impossible to predict where a user might need to browse, a predefined, fixed whitelist invariably blocks users from accessing legitimate web sites. On the other hand, a dynamic whitelist that needs the user's involvement puts a burden on users because, for every site they want to visit, they must first decide whether to put it in the whitelist. This also creates a vulnerability: if a phishing site can convince users to submit sensitive data to it, it may also be able to convince them to put it into a whitelist.

Email authentication [1, 12, 27] prevents phishing by checking if the email is really sent by the party named as the sender. Because phishing emails typically claim to come from the trusted sources, email authentication, once it is widely deployed, can prevent phishing emails from using deceptive addresses. However, the user study in [27] showed that email authentication alone cannot effectively prevent phishing attacks (called "new identity attack" in that study). Moreover, besides emails, attackers can use other channels, e.g., discussion groups or blogs, to lure victims.

Content of the message

Another property amenable to message filtering is the content of the message. Textual content analysis is used widely in antispam and antivirus solutions. Emigh [20] proposed to heuristically analyze the content of email messages and web pages for potentially deceptive links and render them in a way that clearly identify them to users as suspicious. Dangerous messages are detected by searching for well-known patterns, such as spam keywords and virus code signatures. In order to beat content analysis, an attacker can tweak the content to bypass well-known filtering rules. For example, encryption and compression are added to existing viruses in order to bypass antivirus scans. Random characters are inserted into spam emails to enable them to bypass spam filters. Some existing sophisticated phishing attacks used images to display text messages so that they would defeat content analysis.

Spam filtering is one defense that applies at the message retrieval step. Because nearly all phishing attacks are currently launched by spam, getting spam under control may reduce the risk of phishing attacks significantly. Unfortunately, the techniques used by many spam filters, which scan for keywords in the message content to distinguish spam from legitimate mail, are insufficient for classifying phishing attacks, because phishing messages are designed expressly to mimic legitimate mail from organizations with which the user already has a relationship. Even if spam filters manage to reduce the spam problem substantially, phishing is anticipated to move to other transmission vectors, such as anonymous comments on discussion web sites, or narrowly targeted email attacks rather than broadcast spam.

Spoofguard [7] is a Browser Helper Object that calculates a total spoof score for incoming web pages. The calculation is based on common characteristics of known phishing attacks, including:

- Potentially misleading patterns in URLs, such as use of @
- Similarity of the domain name to popular or previously visited domains, as measured by edit distance
- Embedded images that are similar to images from frequently spoofed domains, as measured by image hashing
- Whether links in the page contain misleading URLs
- Whether the page includes password fields but fails to use SSL, as most phishing sites eschew SSL

Sophisticated phishing attacks can tweak their web page contents to bypass these static heuristic rules.

Instead of using the textual content, visual similarity assessment has been proposed [47] to detect phishing websites. In this approach, a suspicious web page is decomposed into a set of salient blocks and compared with the blocks on true web pages stored in a database to see if there is a match. If the matching score is above a threshold, the user is warned. This approach has some potential problems. The first problem is scalability since there are millions of web pages that users can access. Moreover, websites are constantly changing their visual appearance. Thirdly, the attacker may use semantically similar but visually different phishing messages to bypass this protection.

The identity of the sender and the content of the message can be combined together for the message filtering. Cybenko [11] proposed a set of approaches to detect misinformation. For example, the reliability of an incoming message can be determined by a trust rating of the message source, a reliability rating of the message content through collaborations with other redundant information sources, and the linguistic analysis of the message in order to see if the message belongs to the claimed author.

2.1.2 Message presentation

When a message is presented to the user, in either an email client or a web browser, the user interface can provide visual cues to help the user decide whether the message is legitimate.

Current web browsers reflect information about the source and integrity of a web page through a set of visual indicators. For example, the address bar at the top of the window displays the URL of the retrieved web page. A lock icon, typically found in the status bar, indicates whether the page was retrieved through an encrypted, authenticated connection. These cues are currently the most widely deployed and most user-accessible defenses against phishing, and security advisories about phishing warn users to pay close attention to them at all times. [23]

Many proposals for stopping phishing attacks rely on a security toolbar that displays warnings or security-related information in the web browser's interface. Figure 2-5 shows some existing security toolbars.

- SpoofStick [10] displays the website's real domain name, in order to expose phishing sites that obscure their domain name. An attack might use a legitimate-looking domain name as a sub-domain, e.g., www.paypal.com.wws2.us to fool users; SpoofStick would display this domain as wws2.us.

- Netcraft Toolbar [52] displays information about the site, including the domain's registration date, hosting country, and popularity among other toolbar users. This information is thought to be helpful in detecting phishing sites because most phishing sites are short-lived compared to the legitimate sites they imitate, and a large number of phishing sites spoof US-based corporations but are registered in other countries.

- Trustbar [34] makes secure web connections (SSL) more visible by displaying the logos of the website and its certificate authority (CA). This is useful against phishing because many legitimate websites use SSL to encrypt the users sensitive data transmission, but most phishing sites do not. Attackers avoid SSL because obtaining an SSL certificate from a well-known CA, such as VeriSign, requires site identity information that can be traced, and because using a CA that is not known to the browser will trigger a warning and thus might raise the user's suspicion.

- eBay's Account Guard [19] shows a green icon to indicate that the current site belongs to eBay or PayPal, a red icon to indicate a known phishing site found on a blacklist maintained by eBay, and a gray icon for all other sites.

- SpoofGuard [7] calculates a spoof score for the current web page using a set of heuristics derived from previous phishing attacks. It then translates this score into a traffic light: red for spoof scores above a threshold, indicating the page is probably hostile; yellow for scores in the middle; and green for low scores, indicating the page is probably safe.

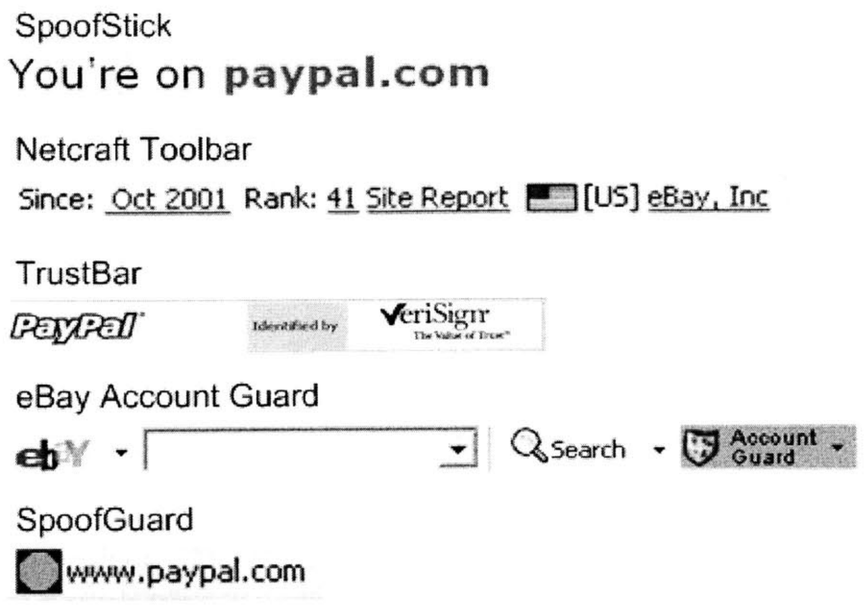


Figure 2-5: Existing security toolbars

There are three types of information displayed by the security toolbars and the browser's visual indicators:

- Neutral information: SpoofStick toolbar and Netcraft toolbar show neutral information about a web site, such as domain name, hostname, registration date and hosting country. The URL displayed in the browser's address bar is also neutral information.
- Positive information: Trustbar shows the positive information at web pages that are SSL protected. The SSL-protected pages are displayed with the site's logo and its CA; a general warning message is displayed for other pages. The SSL lock icon in the browser's status bar and in the Firefox-style address bar fit in this category.
- System decision: eBay's Account Guard and SpoofGuard displays different messages based on their own decision process about whether the site is safe, dangerous, or unknown.

Chapter 3 will discuss in detail why security toolbars cannot effectively prevent

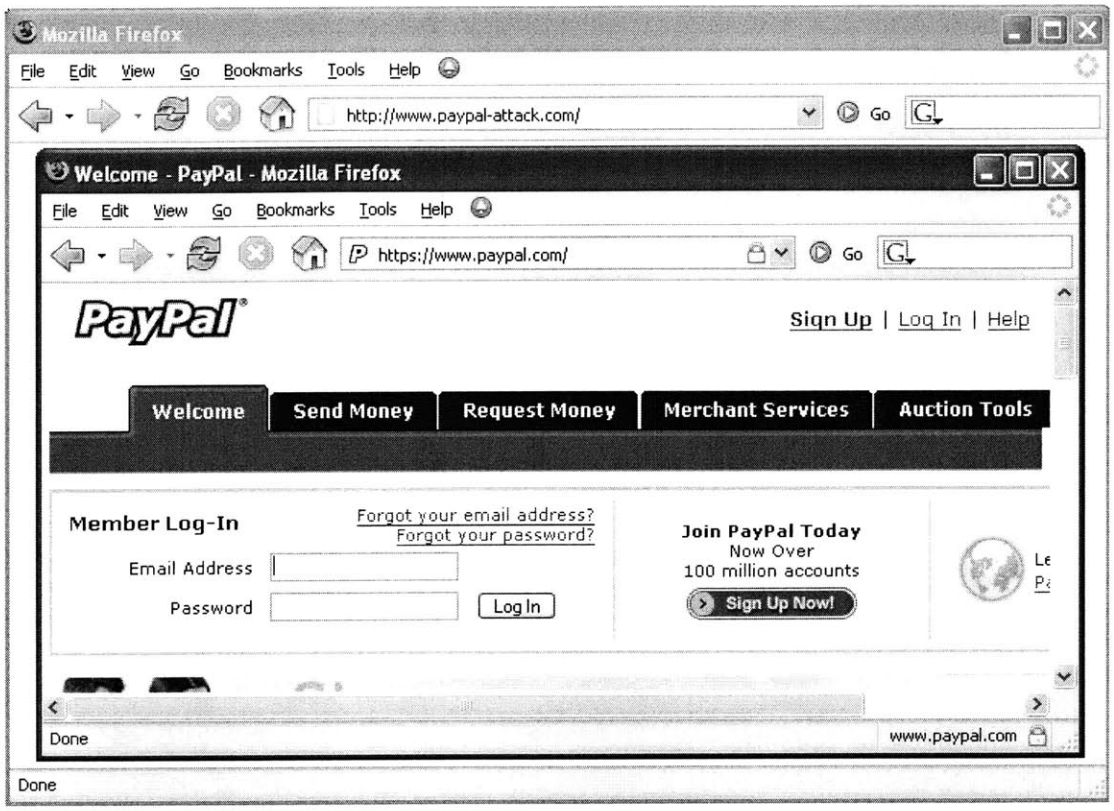


Figure 2-6: Simulated picture-in-picture attack

phishing attacks. Another problem with the toolbars and the browser's visual indicators is that they can be attacked directly by phishing. JavaScript and Java applets have also been used to hide or fake other security cues, including the address bar, status bar, authentication dialog boxes, SSL lock icon, and SSL certificate information. [24, 69, 75]

The *picture-in-picture attack* is a simple attack against visual indicators. A simulated picture-in-picture attack is shown in figure 2-6. In this attack, there is only one Firefox browser window, from www.paypal-attack.com. Inside a simulated "internal" Firefox browser appearing to be at www.paypal.com with the SSL protection is actually a Java Applet. The browser frame and its SSL visual indicators are all faked.

In order to address the problem of faked visual cues, Ye and Smith [74] introduced another type of visual indicator called synchronized random dynamic boundaries. With this approach, all legitimate browser windows change their border colors together at random intervals. Because a spoofed window generated by a remote site has no access to the random value generated on the local machine, its border does not change synchronously with the legitimate window borders. This approach was considered for inclusion in the Mozilla web browser, but this new indicator shares the same usability problem as the other indicators, as will be discussed in Chapter 3.

A related approach, originally proposed by Tygar and Whitten [69], is personalized display, in which legitimate browser windows are stamped with a personal logo, such as a picture of the user's face. The same principle can be used to distinguish legitimate web pages from phishing attacks. For example, Amazon and Yahoo! greet registered users by their names. Anti-phishing advisories (e.g., [18]) suggest that an impersonal email greeting should be treated as a warning sign for a potential spoofed email. PassMark [62] goes even further, by displaying a user-configured image as part of the web site's login page, so that the user can authenticate the web site at the same time that the web site authenticates the user. Dynamic Security Skins [13] proposes to use a personal image to identify the user's master login window and a randomly generated visual hash to customize the browser window or web form elements to indicate successfully authenticated sites. The advantage of this type of approach over the security indicators is that the message display is much harder to spoof and is at the center of the user's attention so that it cannot be easily ignored. The potential problem of this approach is that it still places the burden on users to notice the visual differences between a good site or interface and a phishing one and then correctly infer that a phishing attack is underway.

Another question about personalization is whether the lack of the personalization in a phishing attack would raise sufficient warning flags in a user's mind. The absence of a positive cue like personalization may not trigger caution in the same way that the presence of a negative cue, like a red light in a toolbar, does. Research (e.g., [68]) has shown that it is harder for humans to detect the absence of something, as opposed to its presence. Yee also mentioned this problem as the "Simon Says" problem [76].

2.1.3 User action

Phishing depends on a user not only being deceived but also acting in response to persuasion. As a result, security advisories try to discourage users from performing potentially dangerous actions. For example, most current phishing attacks use email messages as the initial bait, in order to trick the recipient into clicking through a link provided in the email, which points to a phishing server. Security tips [23] suggest that the user should ignore links provided by email, and instead open a new browser and manually type the URL of the legitimate site. This advice is unlikely to be followed. Considering the low frequency of phishing attacks relative to legitimate messages, this suggestion sacrifices the efficiency of hyperlinks in legitimate emails in order to prevent users from clicking misleading links in very few phishing emails.

Two-factor authentication ensures that the user not only knows a secret but also presents a security token when he is logging in. It can prevent phishing because the password that may be captured by the attacker is not the only authentication factor. [22] However, two-factor authentication is a server-side solution. Phishing can still happen at sites that do not support it. Sensitive information that is not related to a specific site, e.g., credit card information and SSN, cannot be protected by this approach either.

Passfaces [54] try to replace the traditional password authentication by letting users correctly choose five face images from five grids of nine photographed faces (figure 2-7). The user's password is not a string of text but a string of faces that are assigned to them during the enrollment stage. Phishing sites that do not know the string of faces that are assigned to a specific user cannot spoof the authentication process to him because there is no correct faces that the user can choose. Passfaces also customize the login page for each individual user. A potential problem for passfaces is the *man-in-the-middle attack* in which the attackers would pass the real face grid to the user and the user's selection to the real site and in the meantime capture the string of faces that are used for authentication.

Yee and Sitaker [78] proposed Passpet as an convenient password manager and

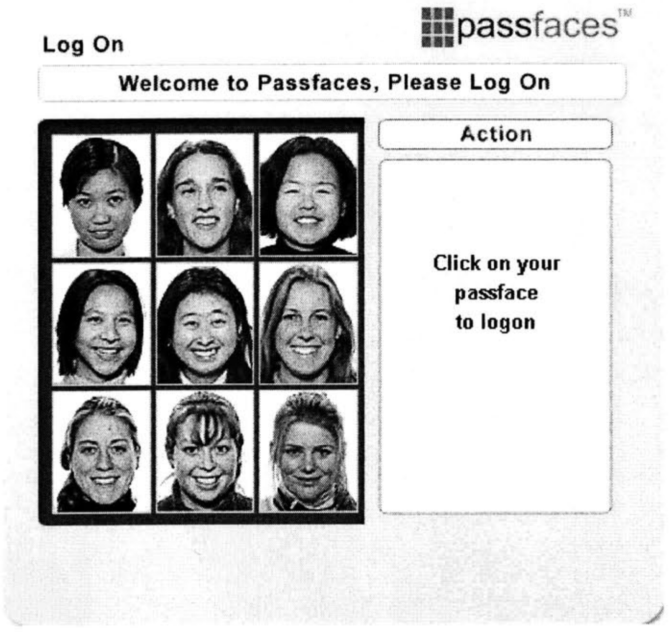

passfaces™
Log On
Welcome to Passfaces, Please Log On
Action
Click on your passface to logon

Figure 2-7: Passfaces authentication (source: Passfaces Corporation)

an anti-phishing solution. With Passpet users do not need to type their passwords at different web sites. They do not even know their real passwords at individual web site. Instead, the web site passwords are automatically generated by the system from the user's master password and the user-assigned site labels. When a user clicks the Passpet icon, the generated password is automatically filled into a detected password field in the current page. This approach prevents users from using weak password and sharing same password among different sites. Moreover, because users do not know the site's password, they cannot leak it even if they are successfully tricked by the phishing attacks. One potential problem of Passpet is that some users may not want their passwords to be hidden from them. Also, Passpet cannot protect other sensitive data, like the credit card information and the bank account information, which are more attractive to the attacker than a password.

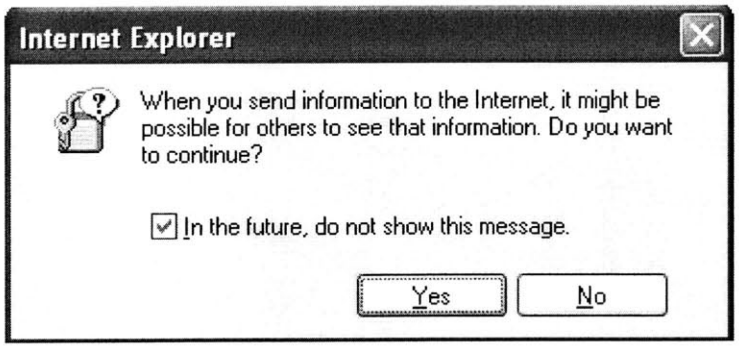
Microsoft InfoCard [6, 39] is an identity meta-system that allows users to manage their digital identities from various identity providers and employ them in different contexts where they are accepted to access online services. Users first register at the identity providers to receive various virtual cards from them. When they go to a web site and are asked for identity data, they click the corresponding virtual card, which will in turn start an authentication process between the current site that the identity provider who issues that card. Again, users do not need to type any sensitive data at all.

The major problem of the InfoCard is that it needs the web sites and the identity providers who support it to add new functionalities. Since InfoCard is a new way for users to provide their identity information, web sites have to be modified to accept the InfoCard submission, by adding an HTML OBJECT tag that triggers the InfoCard process at the user's browser. The sites also have to add backend functionality to process the credentials generated from different identity providers. Moreover, since InfoCard is an identity meta-system, it needs support from various identity providers, including banks that issue bank accounts, credit card companies that issue credit cards, and government agencies that issue government IDs. These identity providers also need to add functionality to process the InfoCard requests. Third, in order to use InfoCard, users have to contact different identity providers to obtain InfoCards from them, which introduces an out-of-band enrollment process between the users and the identity providers.

2.1.4 System operation

In the final step of a successful phishing attack, the user's action is translated into a system operation. This is the last chance to prevent the attack. Unfortunately, because phishing does not exploit system bugs, the system operations involved in a phishing attack are perfectly valid. For example, it is ordinary to post information to a remote server. Warnings (figure 2-8) based solely on system operations will inevitably generate a high rate of false positive errors - that is, warning users about innocent actions. These false-positives eventually cause users to disable the warnings or simply to become habituated to "swatting" the warning away.

A more interesting approach involves modifying the system operation according to its destination. Web password hashing [57] applies this idea to defend against


Internet Explorer

When you send information to the Internet, it might be possible for others to see that information. Do you want to continue?

Figure 2-8: Warning based on system operations

phishing attacks that steal web site passwords. The browser automatically hashes the password typed by the user with the domain name to which it is being sent, in order to generate a unique password for each site - and hence sending useless garbage to a phishing site. Web password hashing assumes that users will type their passwords only into a password HTML element. But this element can be spoofed, and a sophisticated attack may be able to trick users into disclosing their passwords through other channels.

Another novel approach is to monitoring the user's submission in order to detect when a user is entering confidential on a potential phishing site. SpoofGuard [7] stores the hashes of the user's passwords in the user's local machine and monitor the user's online submission to detect if any stored password is being transmitted. If there is a detection, the destination of the submission will be checked to ensure the password is not going to an unauthorized location. One problem is that the monitoring is against HTTP post data. The attacker can easily bypass this monitoring by using a script to scramble the input data before transmitting it. Submission monitoring is ineffective unless it happens at the keystroke level.

2.2 Phishing user studies

Phishing attacks target human users. A number of user studies have been done in order to find out why phishing works and how effective phishing is. In some studies, users were attacked by simulated phishing attacks.

Phishing is an attack that directly targets the human being in the security system. Simulating these kinds of attacks for study purposes raises some special problems. Chief among them is the secondary goal property articulated by Whitten and Tygar [73]: in real life, security is rarely a user's primary goal. The user is primarily concerned with other tasks, such as reading mail, buying a book, or editing a document. Avoiding disclosure of passwords or personal information may be important, but it isn't foremost in the user's mind.

Anti-spam firm MailFrontier Inc did a web survey on how well people can distinguish phishing emails from legitimate ones. The user's goal is to identify five phishing emails from the screenshots of ten emails. Users could not interact with the emails, e.g., clicking their embedded links. About 28% of the time, subjects incorrectly identified the phishing emails as legitimate. [67]

Whalen and Inkpen used an eye tracker to study the user's attention to browser security indicators when doing secure online transactions. Their study [72] found that subjects often looked at the lock icon in the status bar, but rarely clicked on the lock and thus didn't learn anything about the site's certificate. In this study, the subjects were explicitly told to pay attention to the security indicators in the browser to find faking web sites. Security was their primary goal. The experimenter's own passwords and credit card information were used in this study. A web proxy was used to simulate financial transactions with the tested sites.

Dhamija et al. did a study [14] to find out why phishing works. The subjects were shown 20 web sites and asked to determine which sites (12 of them) were fraudulent. The study found that 40% of the subjects made errors, either thinking good sites as fraudulent or thinking phishing sites as legitimate. And 23% of them only paid attention to the web page content but not look at browser-based cues such as the address bar and status bar. Security is again the subject's primary goal.

In April 2005, an interesting study [37] was done at Indiana University Bloomington that showed that social context can make phishing attacks far more effective. The researchers sent out phishing emails to university students, claiming to be from a friend, having mined friendship relations from a social networking site used on campus. The email led to a phishing site that asked for the subject's university username and password. 72% of the subjects provided valid usernames and passwords to the phishing site. In this study, the subject's goal is not security. They were not even told before that they will be attacked. There is only one attack per subject. Because of the study design, the result from this study was expected to faithfully reflect how effective social phishing can be. On the other hand, there were a lot of debates about whether this study should be done in this way. And many subjects felt offended when they eventually knew that they had been attacked.

At least two organizations have initiated phishing attacks against their own members, with the goal of teaching them to protect themselves. [2] The US Military Academy at West Point found that more than 80% of its cadets succumbed to a phishing attack by a fictional colonel. The State of New York mounted two attacks on its 10,000 employees; 15% were spoofed by the first attack, but only 8% by the second, which came three months later.

Chapter 3

User Studies of Anti-Phishing Toolbars

As mentioned in Chapter 2, many anti-phishing proposals rely on a security toolbar that displays security-related information in a web browser. Security toolbars try to bridge the semantic gap between the user's mental model and the system model by showing the system model to the user. Because the toolbars are designed for humans to use, they should be evaluated for usability – that is, whether these toolbars really prevent users from being tricked into providing personal information. Two user studies, described in this chapter, found that security toolbars and visual indicators are ineffective at preventing high-quality phishing attacks. Anti-phishing toolbars, which are located in a peripheral area, unreasonably expect users to constantly pay attention to its message. The toolbar's message shows security information that is not necessary for users to finish their job. And many users cannot correctly interpret the message under phishing attacks in order to notice the danger.

3.1 How to design an anti-phishing user study

A traditional usability test lets human users use the tested software to see if the software can be easily used without introducing any errors. A traditional security test attacks the tested software to see if it can successfully prevent attacks. Since phishing attacks directly target the human users in the security system, and anti-phishing software tries to prevent human users from being tricked, testing anti-phishing software requires combining the above two test schemes together. That is, one must attack users while they are using the tested software, and measure the spoof rate, which is the percentage of successful attacks. There are some issues and tradeoffs that have to be considered in order to correctly design an anti-phishing user study.

3.1.1 Where does the study happen?

A user study can be performed mainly in two ways: a controlled study or a field study. In a controlled study, the subjects are tested in a lab environment controlled by the experimenter in a short period of time, usually less than or about an hour. In a field study, the subjects are tested in their own environment. The testing period can be much longer, often weeks or even months. A controlled study has two main advantages compared to a field study.

- Easy to set up: The tested software does not need to be fully implemented but could be specifically coded for the testing computer system and the given tasks.
- Immediate and adequate feedback: The experimenter can constantly observe the user's behavior during the study. Users can be encouraged to "think aloud" so that the experimenter knows exactly why the users do certain actions. Advanced monitoring equipment, like the eye tracker, can be used to get more information that is hardly obtained by directly observation. The users can be asked for questions about the study immediately after the test when their memory is still fresh.

The biggest drawback of the controlled study, compared to the field study, is that its result does not necessarily reflect the real world situation, since the users are tested in an unfamiliar situation, including the physical environment and the computer system that they are using during the study. Moreover, user behavior may change if they know that they are constantly observed.

3.1.2 What are the user's tasks?

Users can do real tasks during the study, like logging into an account using their own username and password [37]. In most studies, users are given fake tasks. Users can be directly asked to "identify the fake websites (or emails)". But this type of task conflicts with the fact that in real life security is rarely a user's primary goal, as mentioned in section 2.2. A better study design is to give users some tasks to attend to other than security, where the given tasks include some security-related sub-goals that require the subjects to protect a secret from the attack.

Protected secret

If the user has to protect a secret in the study, what should that secret be? The user's own passwords and financial data are one option, but this approach has problems. If the users are told beforehand that they will be *attacked* with their own sensitive data in a study, they may not want to join the study or they may put security – protecting "my" password – as their primary goal. On the other hand, there are ethical and legal concerns about users being attacked, especially with their own sensitive data, without letting them know beforehand.

Another approach is to use data provided by the experimenter. In this situation, the experimenter has to make sure that the users are well-motivated to defend their security in the study. Users will not lose anything even they are successfully attacked, which is different from the outcome under real phishing attacks.

Protected secrets can be passwords or financial data. But most e-commerce transactions with real websites, like buying a book or doing a balance transfer, are hard to revoke. As a result, the transactions in the study have to be simulated. One common solution is to cache the result of the transaction locally and mirror it during the study. Other solutions include using specially-crafted test bank account or credit card numbers provided by a financial institution and setting up a fictional bank site, with which subjects could interact using the fake data.

User's motivation

Users have to be motivated to defend their security in the study. There are in general three ways to do it.

- Motivated by study scenario: Users can be told that they will do some online tasks for some important people, like their boss, in the study. They can also explicitly required to be careful with the sensitive data.

- Motivated by time: Users can be told to finish the given tasks as accurately as possible and if they make mistakes (like falling for attacks) they have to do more tasks.

- Motivated by money: Users can be told to finish the given tasks as accurately as possible, and if they do not make any mistakes they will receive an extra bonus besides the paid fee. However, the motivation should be used carefully because the real money incentive can easily over-motivate users to put security as their primary goal.

It is also very useful to watch for security clues in the study to show that the users do care about the security. For example, users logging out a web site after finishing the given tasks shows that they are concerned about their login session being remembered by the browser and being reused by other people.

3.1.3 What are the attacks?

Users can be attacked in the study by real phishing attacks or simulated ones. The problem with using real phishing attacks is that the study conclusion may no longer be valid when attacks evolve. Simulated attacks use the patterns and the fundamental features of the real attacks, which are less likely to be changed over time. Moreover, any new security mechanism may introduce new vulnerabilities and should also be tested with new potential attacks that target that mechanism.

Attack frequency

In real world, phishing attacks have relatively low frequency. Users may not be attacked for days or weeks. But in a study, user's time should be respected and fully used. Therefore, users have to be attacked intensively. Intensive attacks will make users more alert than their real world practice. The result of the successful attack rate will be a lower-bound of the one in the real world.

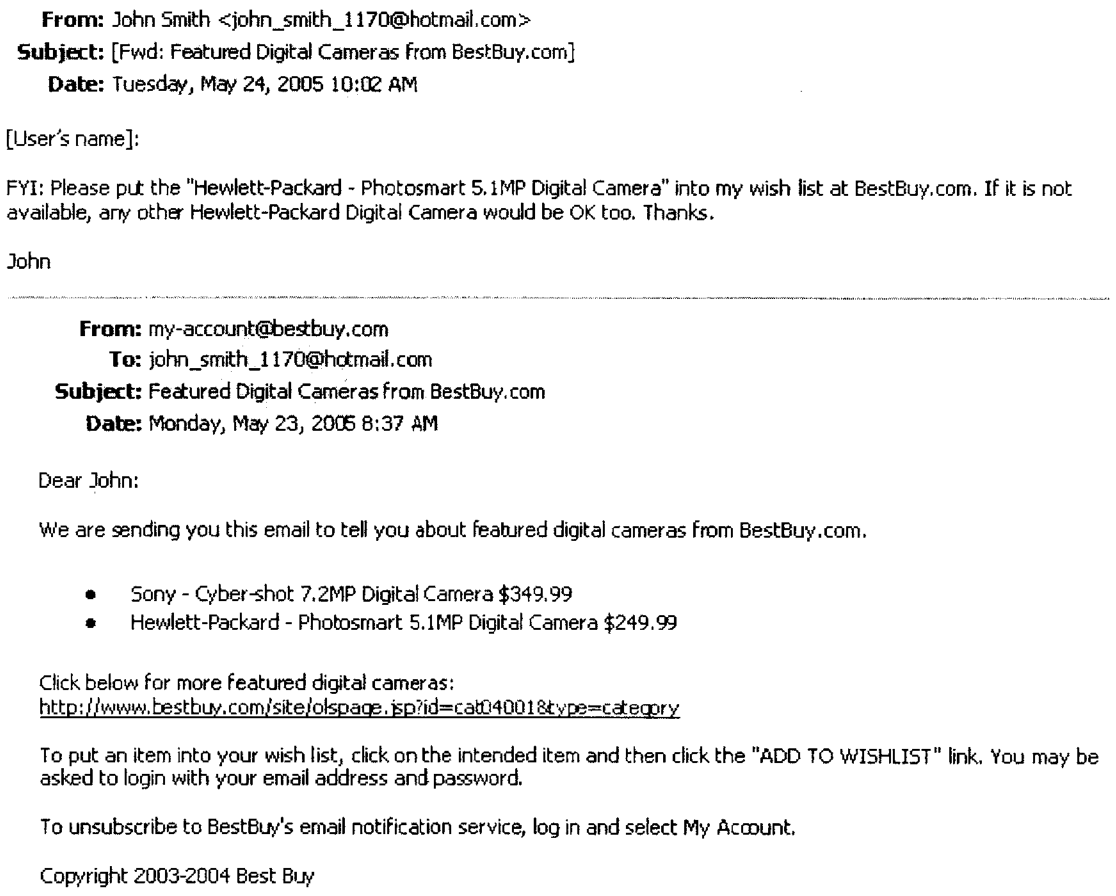
3.2 Study design of the anti-phishing toolbar studies

3.2.1 Study scenario

The anti-phishing toolbar studies are a controlled study. In order to make the security not the user's primary goal, dummy accounts in the name of "John Smith" were set up at various legitimate e-commerce websites. The users played the role of John Smith's personal assistant and were asked to protect his passwords. By only using login information as a secret, a wide variety of existing websites can be used with little setup. Users were given a printout of John's profile, including his fictitious personal and financial information and a list of his usernames and passwords. The task was to process 20 email messages, most of which were the forwarded messages from e-commerce sites. With the forwarded email, John asked the users to manage his wish list at various sites which are protected his username and password. Each message contained a link for the user to click. Figure 3-1 shows a sample message.

3.2.2 Three simulated toolbars

In section 2.1.2, several existing anti-phishing toolbars were mentioned. The studies are not designed for these specific existing toolbars. Instead, they try to find out if the toolbar approach is in general a good way to prevent phishing. Therefore, three simulated toolbars are created (figure 3-2), based on the three types of information



From: John Smith <john_smith_1170@hotmail.com>
Subject: [Fwd: Featured Digital Cameras from BestBuy.com]
Date: Tuesday, May 24, 2005 10:02 AM

[User's name]:

FYI: Please put the "Hewlett-Packard - Photosmart 5.1MP Digital Camera" into my wish list at BestBuy.com. If it is not available, any other Hewlett-Packard Digital Camera would be OK too. Thanks.

John

From: my-account@bestbuy.com
To: john_smith_1170@hotmail.com
Subject: Featured Digital Cameras from BestBuy.com
Date: Monday, May 23, 2005 8:37 AM

Dear John:

We are sending you this email to tell you about featured digital cameras from BestBuy.com.

- Sony - Cyber-shot 7.2MP Digital Camera $349.99
- Hewlett-Packard - Photosmart 5.1MP Digital Camera $249.99

Click below for more featured digital cameras:
http://www.bestbuy.com/site/olspage.jsp?id=cat04001&type=category

To put an item into your wish list, click on the intended item and then click the "ADD TO WISHLIST" link. You may be asked to login with your email address and password.

To unsubscribe to BestBuy's email notification service, log in and select My Account.

Copyright 2003-2004 Best Buy

Figure 3-1: A sample email in the user study

that the existing toolbars display:

- *The Neutral-Information toolbar* shows website information, such as domain name, hostname, registration date and hosting country, as SpoofStick and Netcraft Toolbar do. With this information, users must use their own judgment and experience to decide whether a site is legitimate or phishing.
- *The SSL-Verification toolbar* differentiates sites that use SSL from those that do not. SSL sites are displayed with the site's logo and CA; a general warning message is displayed for other sites. This approach, which imitates Trustbar, seeks to make the user suspicious when a non-SSL page asks for sensitive information such as a password or credit card number.
- *The System-Decision toolbar* displays a red light and the message "Potential Fraudulent Site" if it decides that a web page is actually a phishing attack, an approach that is similar in design to both eBay Account Guard and SpoofGuard. This display is easy for a user to interpret, but it requires the user to trust the toolbar's decision process, which is generally hidden from the user.

3.2.3 Simulated phishing attacks

Five of the 20 forwarded emails were attacks, with links directing the users to a simulated phishing website. Each of these attacks represents a real phishing attack technique that has been recorded by the Anti-Phishing Working Group:

- *Similar-name attack*: Since one way that users authenticate web sites is by examining the URL displayed in the address bar, attackers can use a hostname that bears a superficial similarity to the imitated site's hostname. For example, www.bestbuy.com.ww2.us is used to spoof bestbuy.com.
- *IP-address attack*: Another way to obscure a server's identity is to display it as an IP address, e.g., http://212.85.153.6/ to spoof bestbuy.com.

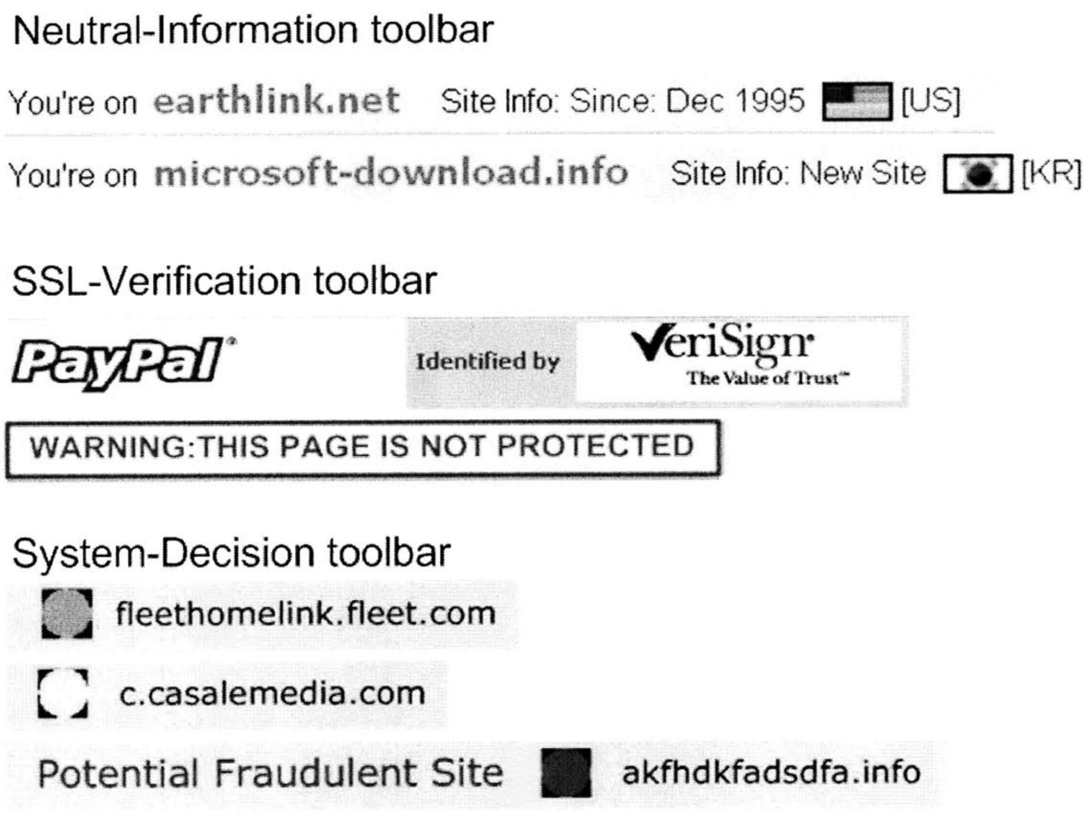

Neutral-Information toolbar
You're on earthlink.net Site Info: Since: Dec 1995 [US]
You're on microsoft-download.info Site Info: New Site [KR]
SSL-Verification toolbar
PayPal
Identified by VeriSign The Value of Trust
WARNING:THIS PAGE IS NOT PROTECTED
System-Decision toolbar
fleethomelink.fleet.com
c.casalemedia.com
Potential Fraudulent Site akfhdkfadsdfa.info

Figure 3-2: Three simulated toolbars that are tested in the study

- *Hijacked-server attack*: Attackers sometimes hijack a server at a legitimate company and then use the server to host phishing attacks. For example, a hijacked site www.btinternet.com is used to spoof bestbuy.com.
- *Popup-window attack*: A popup-window attack displays the real site in the browser but puts a borderless window from the phishing site on top to request the user's personal information. The phishing site displayed the hollywood-video.com site in the browser but popped up a window requesting the username and password. Although this pop-up window lacked an address bar and status bar, it nevertheless included the security toolbar. (figure 3-3)
- *PayPal attack*: The email message warns that John's account has been misused and needs to be reactivated, and points to a phishing website with hostname tigermail.co.kr. This attack requests not only a PayPal username and password, but also credit card and bank account information.

The PayPal attack is different from the other four attacks, which are named *wishlist attacks* because they merely asked the user to log in and modify a wish list. First, the PayPal attack is like current phishing attacks that target online banks and financial services; the wishlist attacks target online retailers instead, which is not as common today, although growing. [25] The PayPal attack is greedy, asking for lots of sensitive information; the wishlist attacks can only steal usernames and passwords. The PayPal attack is far more intimidating, urging users to reactivate their account and threatening to suspend their account if they did not do so immediately. Experienced Internet users were expected to be more suspicious of the PayPal attack.

All three toolbars were configured to differentiate the legitimate sites from the phishing sites. None of the phishing sites used SSL so that the SSL-Verification toolbar always displayed a warning on them. On the System-Decision toolbar, all legitimate sites were displayed as trustworthy (green) but all the phishing sites were displayed as phishing (red) or unsure (yellow). On the Neutral-Information toolbar, the phishing sites and hijacked servers displayed as a "New Site" and some of them were displayed as hosted in other countries outside the US.

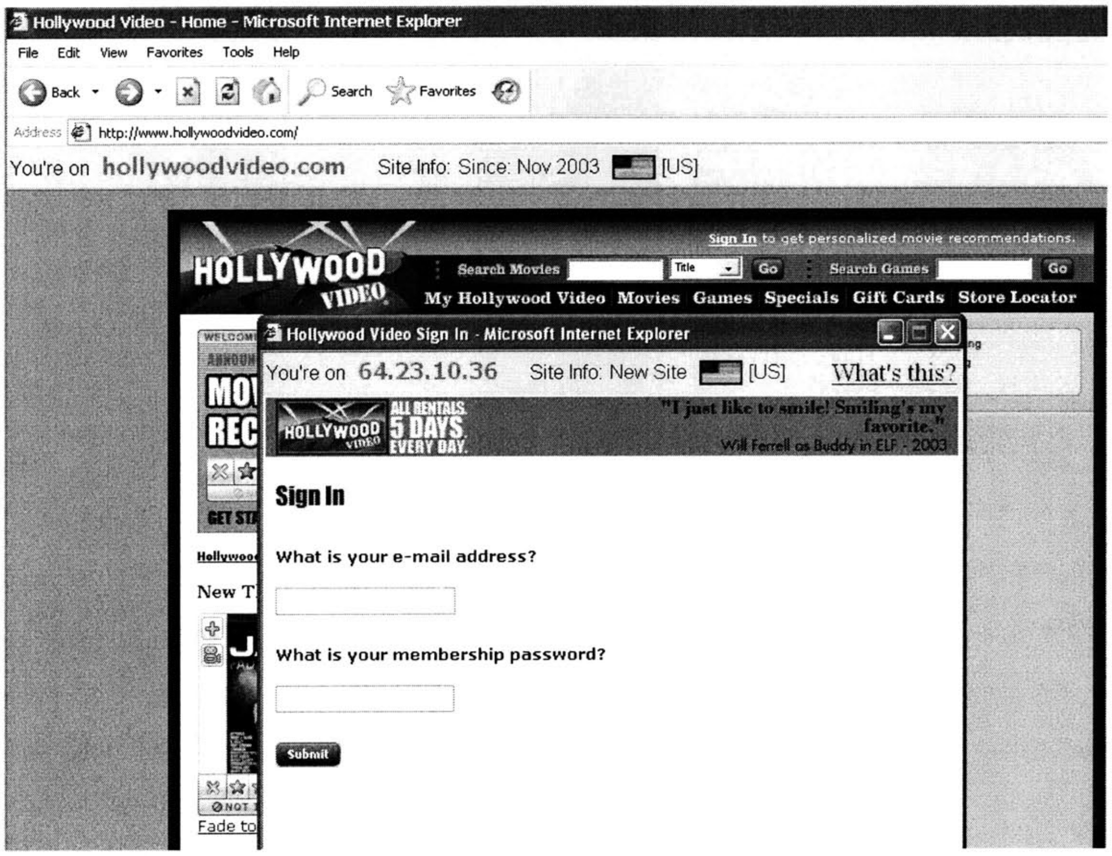

Hollywood Video - Home - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://www.hollywoodvideo.com/
You're on hollywoodvideo.com Site Info: Since: Nov 2003 [US]
Sign In to get personalized movie recommendations.
Search Movies Title Go Search Games Go
My Hollywood Video Movies Games Specials Gift Cards Store Locator
Hollywood Video Sign In - Microsoft Internet Explorer
You're on 64.23.10.36 Site Info: New Site [US] What's this?
ALL RENTALS 5 DAYS EVERY DAY.
"I just like to smile! Smiling's my favorite."
Will Ferrell as Buddy in ELF - 2003
Sign In
What is your e-mail address?
What is your membership password?
Submit

Figure 3-3: The popup-window attack in the study

3.2.4 Toolbar tutorial

Another question in this study design is when and how to give users a tutorial about the security toolbar. Few users read documentation in the real world; some may read an introduction when they download and install a security toolbar, but others may not read anything at all, particularly if the security features are bundled with the web browser.

A pilot study found that the presence or absence of a tutorial has a strong effect on performance. When five pilot subjects received a printed tutorial explaining the security toolbar, showing how it looked for both legitimate and phishing websites, only 1 out of 15 attacks (7%) was successful. Another six pilot subjects received no printed tutorial; instead, a "What's this?" link was added in each toolbar which displayed the tutorial in a popup window. These subjects succumbed to 17 out of 18 attacks (94%); no subject clicked the "What's this?" link.

This result was problematic. In the former case, the printed tutorial gave the pilot subjects too strong a clue that security was the primary goal in the study. In the latter case, subjects had no idea what the security toolbar meant, or its role in preventing phishing attacks.

Based on this experience, the tutorial was introduced as part of the scenario. In the experiment, John Smith forwards to the subject an email from his company's system administrator. The email says that a security toolbar has been installed on the company's computers to prevent phishing attacks. The message contains a link to the tutorial. When John Smith forwarded this email to the subject, he explicitly requests that they be careful with his personal information. Figure 3-4 shows the tutorial of the Neutral-Information toolbar and how it was introduced to the subjects.

The tutorial email appeared in the middle of the study, as the 11th of the 20 emails, where it could serve as a control to see how users behaved before and after seeing the tutorial. The PayPal attack was the 10th email because of its uniqueness. The remaining four attacks occurred at the 5th, 8th, 16th and 19th emails, with each type of wishlist attack randomly assigned to one of these four positions. These fixed positions were chosen to space out the attacks.

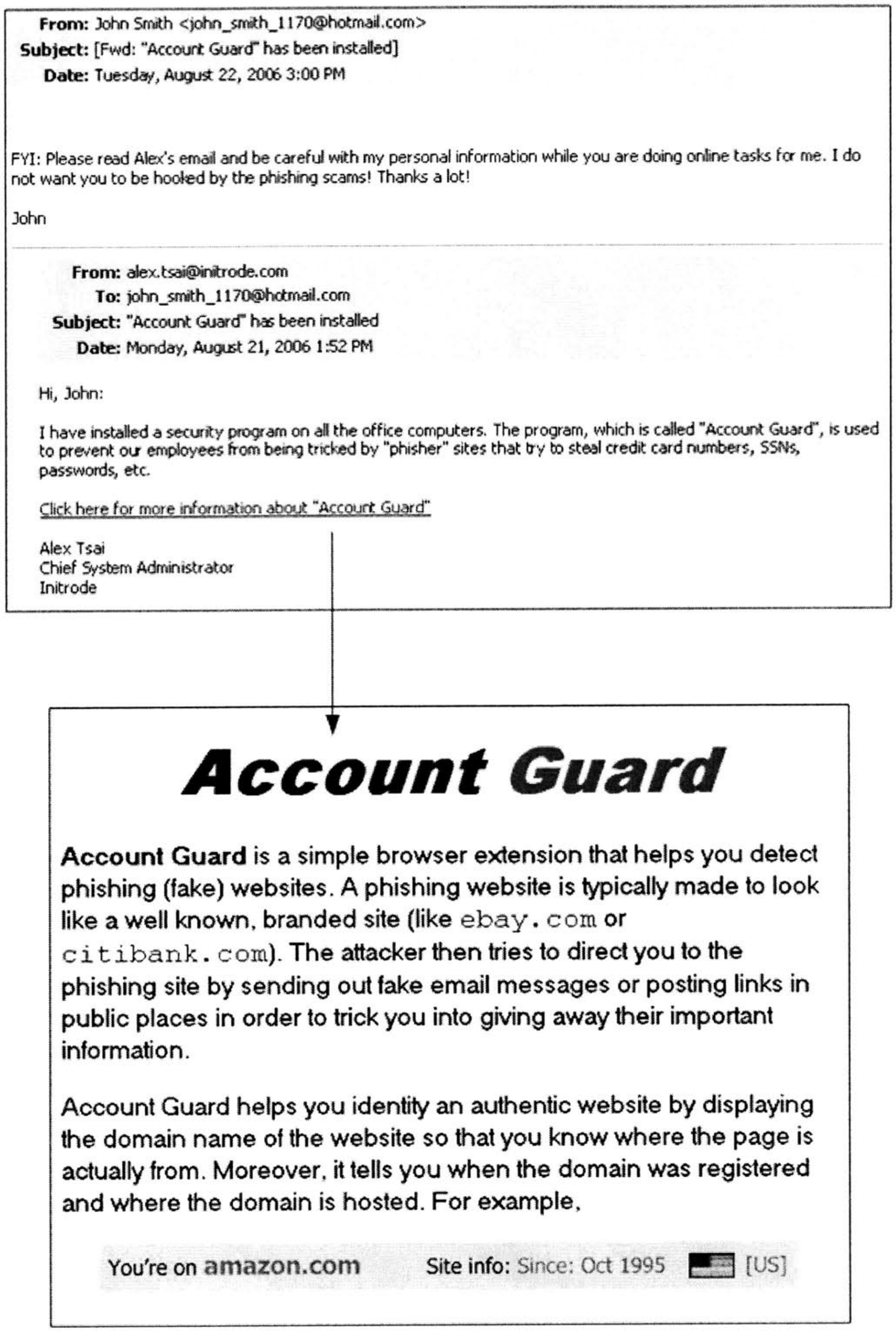
3.3 User study implementation

In order to simulate attacks against users, I needed to completely control the display of the toolbars and other security indicators. Users in the study interacted with a simulated Internet Explorer built inside an HTML application running in full screen mode (figure 3-5). Different HTML frames displayed different browser components, including the security toolbars. The locations and sizes of the toolbars were consistent with the existing toolbars that they are based on. The Neutral-Information toolbar and the System-Decision toolbar were located below the address bar and above the main browsing window. The SSL-Verification toolbar was located below the title bar and above the menu bar. The address bar took the Firefox approach by using the yellow background and a lock icon to indicate SSL connections. The status bar displayed a lock icon for SSL connections.

This study simulated ideal phishing attacks whose content is a perfect copy of the actual website. (This is realistic, since an attacker might not bother mirroring the entire site, but might simply act as a man-in-the-middle between the user and the real site.) As such, the main frame in the browser always connected to the real website, regardless of whether the site was supposed to be phishing or not. To simulate phishing attacks, the appearance of the HTML frames that displayed the browser's security indicators - including the security toolbar, the address bar and the status bar - was changed to indicate that the web page was served by an unusual source, e.g., tigermail.co.kr rather than paypal.com.

3.4 Study participants and protocol

This study was done in March 2005. A total of 30 subjects with previous experience in online shopping, 14 females and 16 males, were recruited by online and poster advertising at MIT. Twenty subjects were college students from 10 different majors.



From: John Smith <john_smith_1170@hotmail.com>
Subject: [Fwd: "Account Guard" has been installed]
Date: Tuesday, August 22, 2006 3:00 PM

FYI: Please read Alex's email and be careful with my personal information while you are doing online tasks for me. I do not want you to be hooked by the phishing scams! Thanks a lot!

John

From: alex.tsai@initrode.com
To: john_smith_1170@hotmail.com
Subject: "Account Guard" has been installed
Date: Monday, August 21, 2006 1:52 PM

Hi, John:

I have installed a security program on all the office computers. The program, which is called "Account Guard", is used to prevent our employees from being tricked by "phisher" sites that try to steal credit card numbers, SSNs, passwords, etc.

Click here for more information about "Account Guard"

Alex Tsai
Chief System Administrator
Initrode

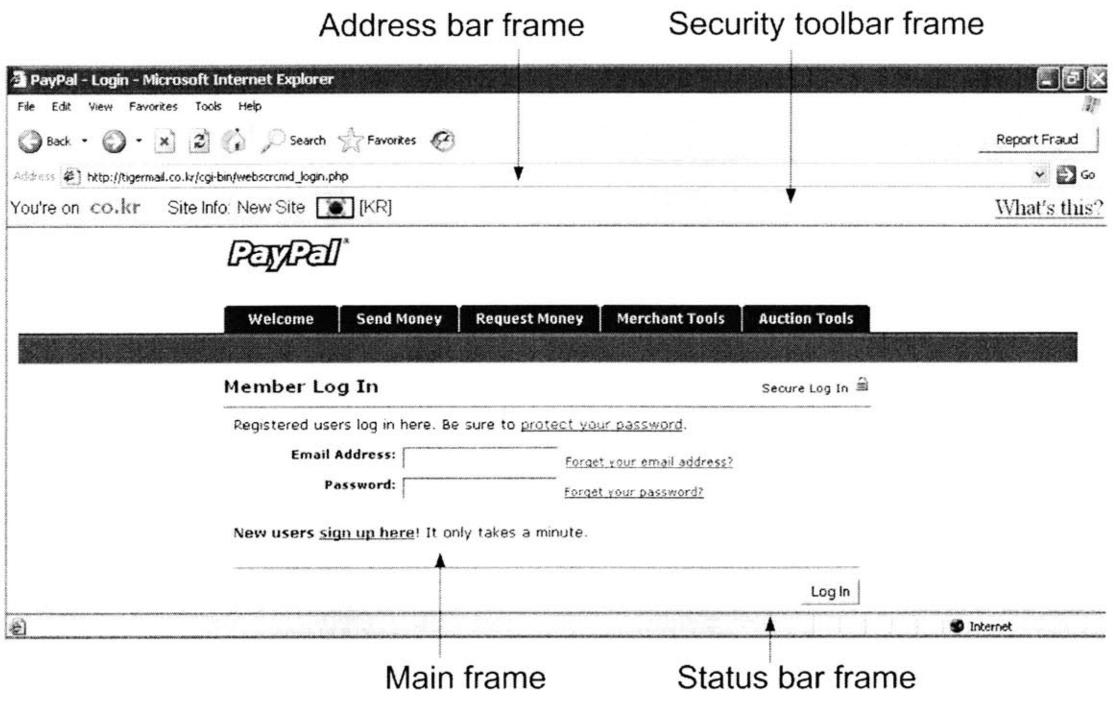
Account Guard

Account Guard is a simple browser extension that helps you detect phishing (fake) websites. A phishing website is typically made to look like a well known, branded site (like `ebay.com` or `citibank.com`). The attacker then tries to direct you to the phishing site by sending out fake email messages or posting links in public places in order to trick you into giving away their important information.

Account Guard helps you identity an authentic website by displaying the domain name of the website so that you know where the page is actually from. Moreover, it tells you when the domain was registered and where the domain is hosted. For example,

You're on amazon.com Site info: Since: Oct 1995 [US]

Figure 3-4: Tutorial of the Neutral-Information toolbar in the study



Registered users log in here. Be sure to protect your password.

New users sign up here! It only takes a minute.

Figure 3-5: Browser simulation using an HTML application

All subjects had at least college education. The average age was 27 (the age range was from 18 to 50). Each of the three security toolbars was tested on 10 subjects.

In order to gauge subjects' experience with online shopping, they were asked which of the 19 selected e-commerce sites they had visited. All 30 subjects had used Amazon, and 25 or more had used PayPal, Travelocity, Bestbuy, and Yahoo. On average, each subject had used 10 of the sites in this study.

Before the study, subjects were given a consent form which (1) explained that the purpose of the study was to test web browser security indicators that detect fake web pages that look like pages from well-known legitimate websites; (2) indicated that the purpose of these fake websites is to trick people into making dangerous decisions or taking dangerous actions; and (3) encouraged the subjects to detect all the fake web pages and report them by clicking the "report fraud" button in the browser's toolbar. All the subjects were required to read the consent form carefully, especially the study procedure part.

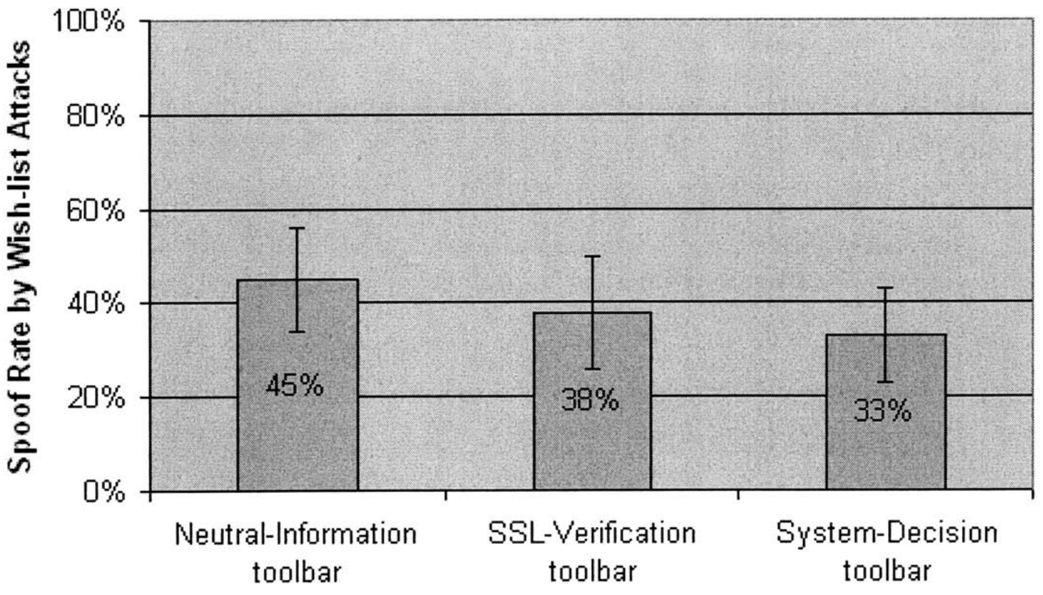


Figure 3-6: Spoof rates with different toolbars

After the consent form, the subject was briefed about the John Smith scenario and their role as John Smith's assistant. This briefing did not mention security at all. I personally observed the subjects' browsing behaviors during the study. I did not interrupt the study except when subjects clicked the "report fraud" button, at which point I asked them to explain why they reported fraud.

3.5 Results and discussion

Spoof rate is the fraction of simulated attacks that successfully obtain John's username and password or other sensitive information without raising the subject's suspicion.

Figure 3-6 shows the spoof rates of wishlist attacks for each toolbar. These spoof rates, 45% for the Neutral-Information toolbar, 38% for the SSL-Verification toolbar, and 33% for the System-Decision toolbar, are all significantly higher than 0%, the ideal. No significant difference was found between the toolbars by a single-factor ANOVA test. But this hardly matters since all the toolbars have high spoof rates.

Among the 30 subjects, 20 were spoofed by at least one wishlist attack (7 used the Neutral-Information toolbar, 6 used the SSL-Verification toolbar, and 7 used the System-Decision toolbar). These subjects were interviewed at the end of the study to find out why they did not recognize the attacks:

- 17 subjects (85%) mentioned in the interview that the web content looked professional or similar to what they had seen before. They were correct because the content was the real web site, but a high-quality phishing attack or man-in-the-middle can look exactly like the targeted website as well. Seven of these subjects were observed to use security-related links on the site itself to decide if a site was legitimate or not – for example, clicking on the VeriSign seal, the site's privacy policy, contact information, copyright information, or a credit card security claim. Of course, attackers can and do fake these indicators. A lot of research has been done to improve web credibility (e.g., [44]), and attackers have clearly adopted these techniques.

- 12 subjects (60%) used rationalizations to justify the indicators of the attacks that they experienced. Nine subjects explained away odd URLs with comments like:

  - *www.ssl-yahoo.com is a subdirectory of Yahoo!, like mail.yahoo.com.*
  - *sign.travelocity.com.zaga-zaga.us must be an outsourcing site for travelocity.com.*
  - *Sometimes the company [Target] has to register a different name [mytargets.com] from its brand. What if target.com has already been taken by another company?*
  - *Sometimes I go to a website and the site directs me to another address which is different from the one that I have typed.*
  - *I have been to other sites that used IP addresses [instead of domain names].*

  Four subjects explained away the popup window that asked for a username and password. One subject commented that she must have triggered the popup window herself, by clicking "Register for new account" instead of "Sign in for existing account".

One subject explained away a toolbar message showing that Yahoo! was a "New Site" and located in Brazil by reasoning that Yahoo must have a branch in Brazil. Another explained away the warning on the System-Decision toolbar by saying that it was triggered because the web content is "informal," just like a spam filter says that "this email is probably a spam."

- Nine subjects (45%) said that the reason they were spoofed was that they were focused on finishing the study tasks, i.e., dealing with John Smith's email requests. Three explicitly mentioned that, although they noticed the security warnings, they had to take some risks to get the job done. Simply warning these subjects that something is wrong was not sufficient: they needed to be provided with a safe alternative way to achieve their goals.

- Five subjects (25%) claimed that they did not notice the toolbar display at all for some attacks.

- One subject extensively clicked links on the web pages to test whether the web site worked properly. By relying on the site's behavior as an indication of its authenticity, this subject was fooled by all of the wishlist attacks.

As shown in figure 3-7, the similar-name attack had the highest spoof rate, 50%, among all simulated phishing attack techniques. But no matter how the phishing URL is presented, the spoof rates are always high, with 43% for the hijacked-server attack and 33% for the IP-address attack. The popup-window attack had a relatively low spoof rate of 27% - many subjects thought that using the popup window for login information was abnormal and suspicious. The spoof rate differences were not significant by a single-factor ANOVA test.

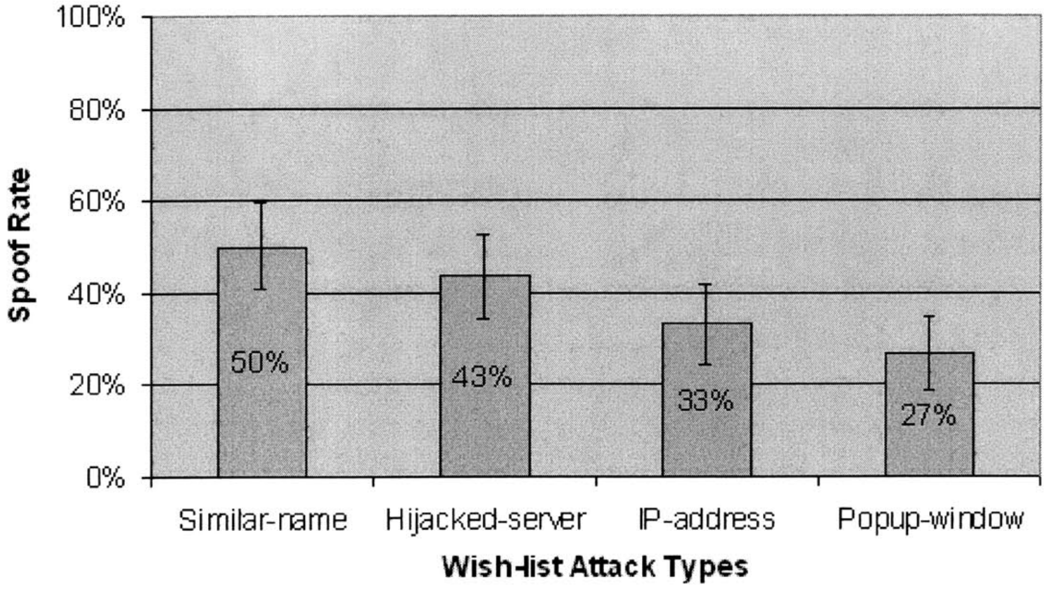


Figure 3-7: Spoof rates of wishlist attacks with different attack types

3.5.1 Learning effect

Seven subjects clicked the toolbar's "What's this?" link before the tutorial email (1 using the Neutral-Information toolbar, 2 using the System-Decision toolbar, and 4 using the SSL-Verification toolbar.) Subjects using the SSL-Verification toolbar clicked "What's this?" even before the first attack. I believe that this is because the toolbar displayed a warning message on all pages that did not use SSL, which is the case for many web pages.

Difference found in spoof rates for wishlist attacks before and after the subjects saw the tutorial, either by clicking the "What's this?" link or by reading the tutorial email, is statistically significant (one-tail t(43) = 2.27, p = 0.014). Figure 3-8 shows the spoof rate before the tutorial was 52%, while after the tutorial it dropped to 26%. Although a decrease was found with all three toolbars, the decrease was significant for the Neutral-Information toolbar (one-tail t(18) = 1.84, p = 0.04), marginally significant for the System-Decision toolbar (one-tail t(12) = 1.52, p = 0.077), and

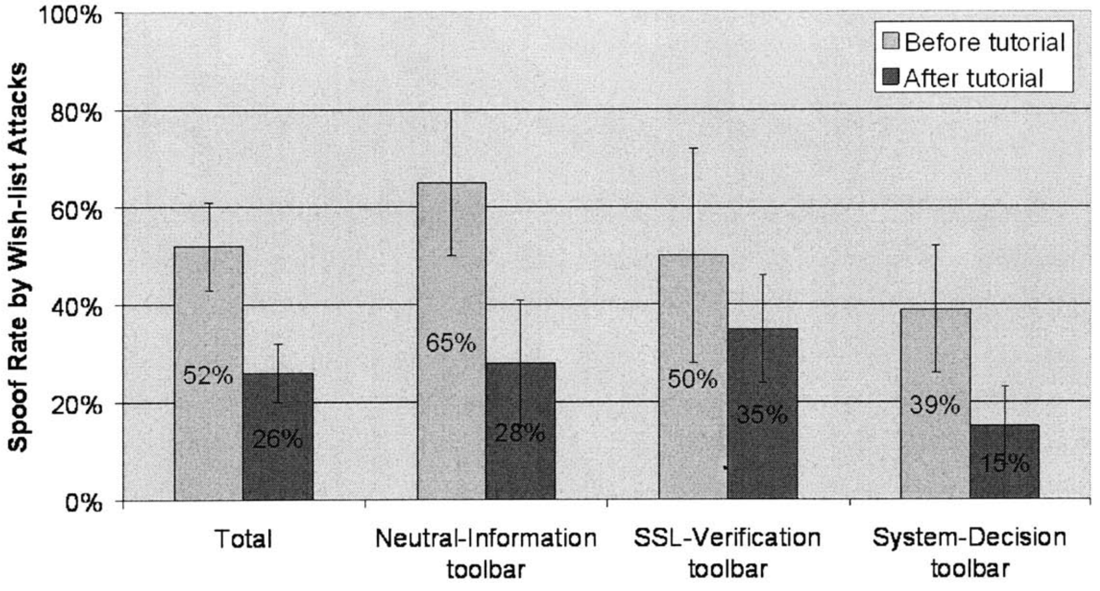


Figure 3-8: Spoof rates before and after the tutorial

not significant for the SSL-Verification toolbar (one-tail t(7) = 0.61, p = 0.28).

Several subjects mentioned that the tutorial email helped them to pay more attention to the security toolbar and better understand its display, explaining the drop in spoof rate following the tutorial.

Subjects using the Neutral-Information toolbar and the System-Decision toolbar saw their spoof rates significantly drop following the tutorial. This was not true of subjects using the SSL-Verification toolbar. One explanation is that the toolbars had different levels of accuracy. I tried to make every toolbar accurate enough to distinguish phishing sites from legitimate sites. The System-Decision toolbar displayed a red or yellow light at the phishing sites but a green light at the good sites. The Neutral-Information toolbar showed all phishing sites as either a "new site" or hosted in a non-US country (or both), but all good sites as hosted in the US and in existence for several years. But it turned out that 9 of the 18 online stores chosen for this study had login pages that were not protected by SSL, so the SSL-Verification toolbar produced warnings even for legitimate sites. Thus, the SSL-Verification toolbar failed to

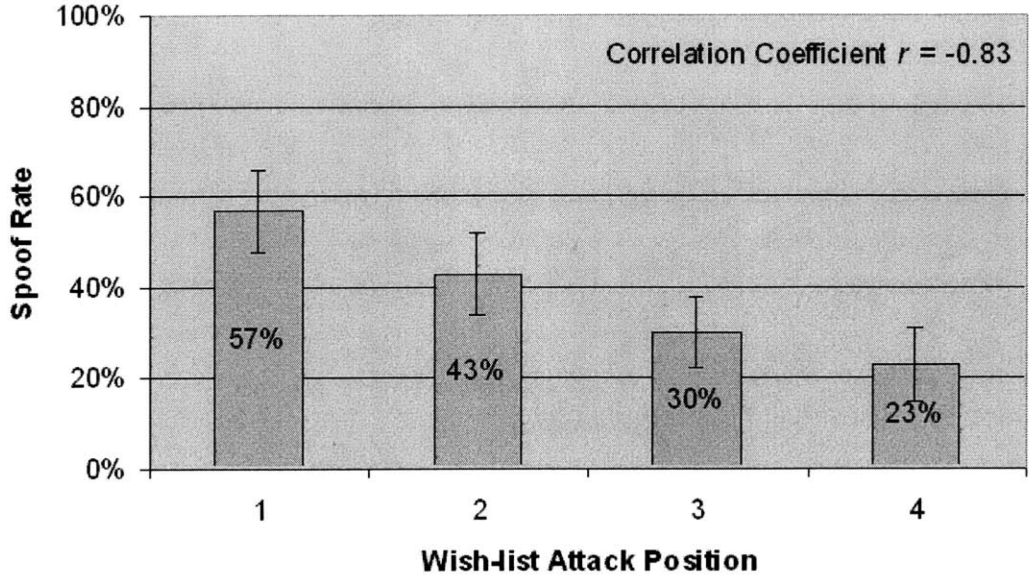


Figure 3-9: Spoof rates of wishlist attacks at different attack positions

adequately distinguish fake sites from good ones.

The tutorial was not the only factor affecting subjects' learning, of course. Another contribution to the decrease in the spoof rate before and after the tutorial is that the spoof rate steadily decreased for each attack as the subjects experienced more wishlist attacks and learned how to detect them, as shown in figure 3-9.

3.5.2 Security awareness

As discussed before, one of the risks of using an artificial scenario is that users may not care about the fictional John Smith's security at all. In this study, a number of indicators were found to show the subjects were behaving as if they did care about the security of John Smith's accounts. Designing these kinds of secondary indicators into a security study turned out to be a good idea.

For example, 18 subjects unchecked the "Remember Me" checkbox on the login page of at least one site. This checkbox, which is generally checked by default and must be explicitly unchecked, controls whether John Smith's login information is recorded in a cookie. Furthermore, 13 subjects explicitly logged out or tried to log out of at least one web site after finishing a task. These cautious subjects (23 in all) were protective of John Smith's online identity and did not want the browser to remember the login sessions. The subjects were never told anything about unchecking "Remember Me" or logging out. The other 7 subjects also exhibited suspicion and caution at least once in the study, either by reporting fraud or by trying to carefully explore a web site to determine if it was legitimate.

Subjects also demonstrated caution with false alarms - believing that a good site was an attack. Subjects did not finish tasks at good sites 3.3% of the time (13 out of 390 tasks) because of security concerns. There were six false alarms before the tutorial and seven after the tutorial. False alarms were generally due to browser warnings generated by the legitimate site (such as "You are about to be redirected to a connection that is not secure").

3.5.3 The PayPal attack

As discussed above, the PayPal attack is very different from the wishlist attacks. The difference is reflected in the study. The PayPal attack had a significantly lower spoof rate (17%) than the wishlist attacks (38%) (one-tail t(56) = -2.63, p = 0.005), as shown in figure 3-10. Ten subjects said that they had seen similar phishing emails in the real world, so they could detect the PayPal attack just by reading the email message, without even clicking through to the phishing site. The wishlist attacks have a lower spoof rate (28%) on these 10 subjects than the other 20 subjects (44%). But the difference is not significant (one-tail t(23) = -1.36, p = 0.09). Some subjects did not feel comfortable providing John Smith's credit card and bank account information, and eventually noticed the suspicious signs from the toolbar or the suspicious URL from the address bar and thus avoided the attack.

However, there were still five subjects out of 30 (17%) who were tricked by the PayPal attack (at least one using each toolbar). Four were PayPal users in real life. They were spoofed because the content of the site looked authentic. One typical

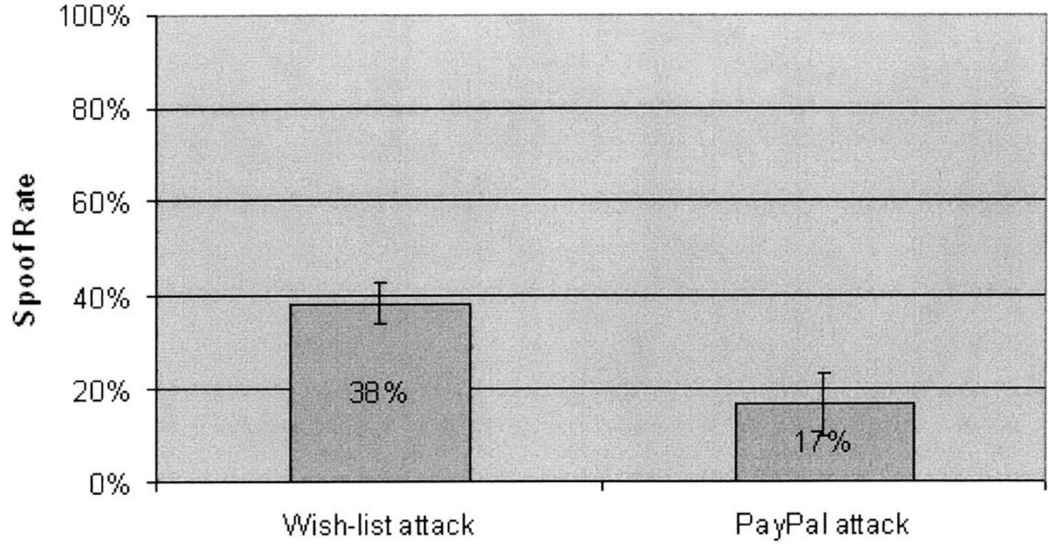


Figure 3-10: Spoof rates between the wishlist attack and the PayPal attack

comment was "I've used PayPal before and this site looks exactly the same. If I trust a site from my experience, I am not suspicious." They also justified the request as being reasonable. One subject said that "they need this information [the credit card and the bank account information] to charge me." Thus, familiar phishing attacks can continue to be persuasive and effective, even with security toolbars to warn the user.

3.5.4 Population sampling bias

Because the subjects were recruited at MIT, they are likely to be more technical savvy than the majority of the Internet users. (All subjects have at least college education.) Subjects who knew more about phishing did a better job in detecting phishing attacks than those who knew less (e.g., in the PayPal attack). As a result, figure 3-6 is probably a lower bound of the spoof rates if this study were done with ordinary Internet users.

3.5.5 Subjective ratings and comments on toolbars

Subjects were asked at the conclusion of the study to rate the effectiveness of the address bar, status bar, the security toolbar that they used in differentiating authentic web sites from phishing sites, on a scale from -2 (very ineffective) to 2 (very effective). Figure 3-11 shows the mean ratings.

Among the three toolbars, the SSL-Verification toolbar was rated as less effective, although the difference was not significant. One reason might be because the SSLVerification toolbar could not distinguish phishing sites from legitimate sites that do not use SSL. Sadly, many such sites exist in the wild, and some were used in this study. But even when the toolbar functioned properly, it was often ignored. One subject commented that the toolbar looked like an advertisement banner, so it was unclear whether it was put there by the browser or by the site.

The other two toolbars were thought more effective than the browser's own address bar. A common remark on the security toolbar was that the toolbar worked with the address bar: the toolbar alerted and warned the subject, causing the subject to pay more attention to the address bar.

Some subjects did not know how to interpret the information the toolbars displayed – especially the Neutral-Information toolbar. One subject said: "How do I have any idea about the [registration] time and location of a site?"

3.6 Follow-up study

A more effective interface for getting the user's attention about a phishing web site is to actually block access to it – for example, by popping up a modal dialog box when the site is visited. Several security toolbars, including Netcraft Toolbar, eBay Account Guard and SpoofGuard, display a pop-up warning when they have high confidence that the current site is phishing. This warning is likely to get the user's attention since it appears in the center of the browser and impedes progress until it is acknowledged.

A pop-up is a very aggressive warning, disrupting the user's task, so it must

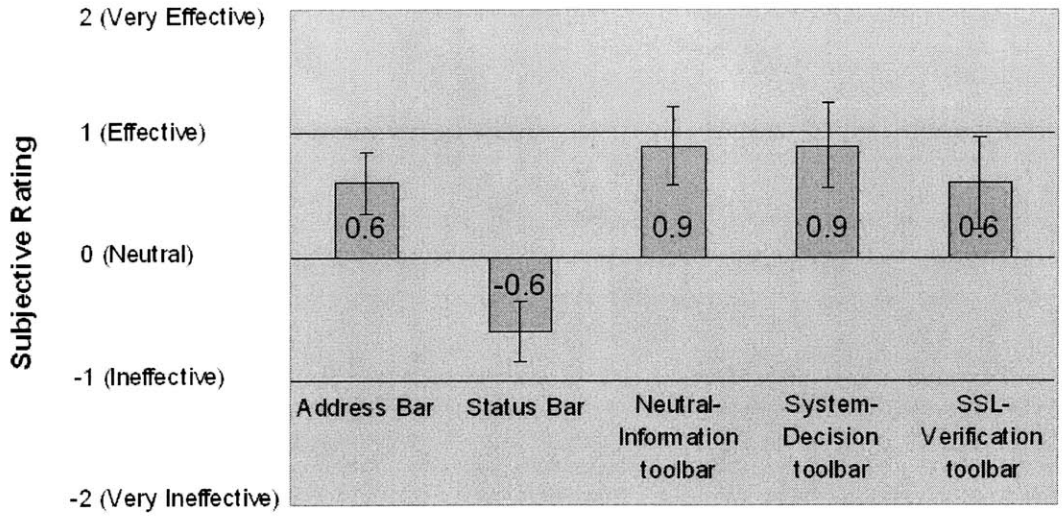


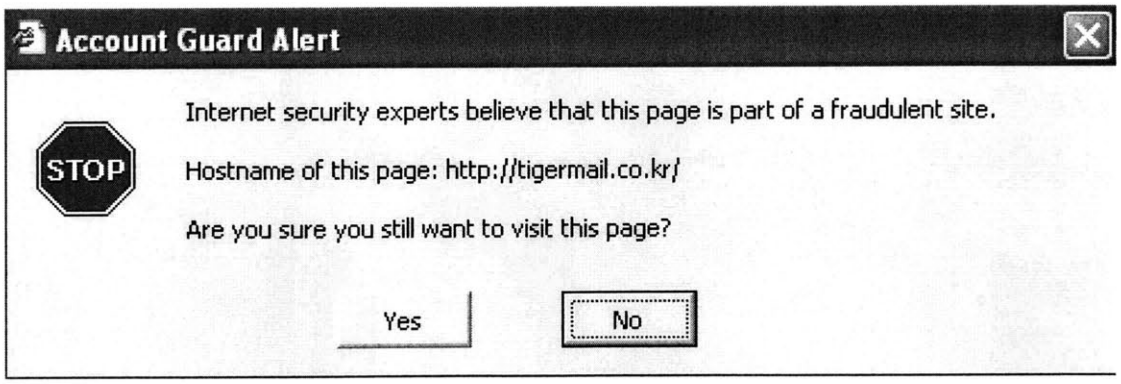
Figure 3-11: Subjective ratings of the address bar, the status bar and the toolbars

be accurate or it will be disabled. Since phishing attacks evolve rapidly, security toolbars are rarely certain enough about a new phishing attack to display a pop-up. As a result, these toolbars depend more heavily on the persistent toolbar display to warn users about new dangers. This is why the first study focused on the toolbar display.

Nevertheless one might expect a pop-up dialog to be more effective at prevent phishing attacks. To find out, a follow-up study with new subjects was run to test the pop-up alert technique. The second study used the same scenario and the same attacks with the same numbering and positioning of attacks. Half of the subjects saw a blocking warning box (figure 3-12) at the phishing sites, which closely resembles the warning used by the Netcraft Toolbar. The rest acted as a control, using only a standard browser interface with the address and status bars are the main security indicators.

The follow-up study had 20 subjects aged 19 to 37 (average age 23). 18 (90%) were college students, 13 were males. It was done in July 2005, four months after the previous study.

It turned out that these subjects exhibited more caution than the subjects in the


Internet security experts believe that this page is part of a fraudulent site.

Hostname of this page: http://tigermail.co.kr/

Are you sure you still want to visit this page?

Figure 3-12: A sample blocking warning box at a phishing site

first study. Their degree of caution was measured as a sum of three indicator variables: 1 point if the subject ever tried to log out of a site; 1 point if the subject unchecked "remember me" at a login page; and 1 point if the subject failed to finish the task at a good site because of security concerns. The second study's subjects had an average caution score of 2.0, compared to 1.3 for the first study, a significant difference (two-tail t(39) = -3.29, p = 0.002). Possible reasons for the difference include the demographics of the subjects and substantial media coverage about phishing and identify theft in the intervening four months.

As expected, the blocking warning box dramatically decreased the spoof rate of the wishlist attacks in the study, as shown in figure 3-13. The decrease was statistically significant (one-tail t(9) = -2.88, p = 0.01). Seven out of the 10 subjects with the regular browser interface were spoofed by at least one wishlist attack:

- Six subjects (86%) said that the web content looked good or the same as they had seen before.
- Two subjects (29%) rationalized the suspicious URL. One subject, experiencing a similar-name attack at www.walmart.com by www.walmart.com.globalupdate2.com, said that "global-update2 is a service to do the website's global updating and this service is working for Walmart."
- Three subjects (43%) did not look at the URL in the address bar at all. One

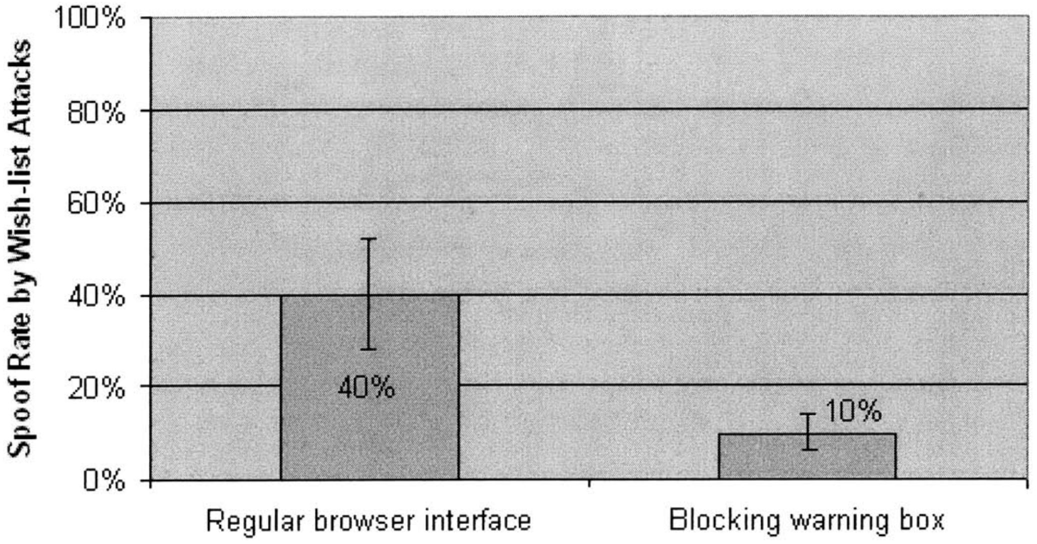


Figure 3-13: Spoof rates with a regular browser interface and the blocking warning box

said that "I did not bother to look at the address bar since the page looked so good."

Two subjects with the regular browser interface were spoofed by the PayPal attack, both PayPal users in real life. They mentioned that the site looked just like PayPal's and that the phishing email was a reasonable request by PayPal.

The results from the 10 subjects who used the regular browser interface supported the conclusions from the first user study: many users depend on the web content to authenticate the site's identity. Even though they are cautious and notice suspicious signs from the browser's security indicators, since these signals are weak compared to the strong signals from convincing web content, the users tend to ignore or explain away the security indicators.

Of the 10 subjects who used the blocking warning box, none were spoofed by the PayPal attack but four were spoofed by wishlist attacks:

- None of the four spoofed subjects offered that the content of the page was convincing as a reason that they were spoofed - somewhat ironic, since the content was in fact the real site! Apparently the warning box blocking the page is a stronger signal than the web content.

- Two subjects believed the warning and knew the site was phishing, but still wanted to complete the task. The subjects claimed that a wish list was not sensitive enough for them to decline John's request. Apparently they did not realize that revealing John's shopping password to an attacker could result in financial loss if John used the same username and password at another site.

- The other two subjects did not trust the blocking warning box. One said that he had never seen such a warning before. The other thought that the warning was wrong. This subject had his own anti-phishing strategy: he typed a wrong password at the suspicious site's login page. If the site accepted the wrong password, he inferred that the site was phishing. But if the site rejected the wrong password, he concluded that the site was good and that the warning box was making an error. Clearly, this strategy does not work against phishing sites executing man-in-the-middle attacks, for these sites pass usernames and passwords on to the real site to perform the validity check. Interestingly, two other subjects in the follow-up study also used this same strategy to check a site's authenticity; this behavior was never seen in the first study.

The follow-up study confirms that many users do not know how sophisticated a phishing attack can be. Some subjects were impressed by the simulated phishing attacks. One typical comment: "I cannot imagine that an attacker can make the attack so elegant to mirror a whole site." Others wrongly believed that phishing sites cannot check password validity since they don't have the correct password.

3.7 Why don't the security toolbars work?

Security toolbars try to bridge the semantic gap between the user's mental model and the system model by showing the system model to the user. I evaluated three types of security toolbars, as well as browser address and status bars, to test their effectiveness at preventing phishing attacks. All failed to prevent users from being spoofed by high-quality phishing attacks for the following reasons:

- Toolbars and security indicators located in a peripheral area provide a much weaker signal than the centrally displayed web page and can be easily overwhelmed by the convincing web content. Many users relied on the web content to decide if a site is authentic or phishing. In these two studies, simulated phishing sites had high-fidelity content. As a result, even though the security toolbar tried to alert the subjects, many of them disregarded the warning because the content looked so good. Confirmation bias also makes the toolbar less credible. When a user clicks an email link to a web site, he believes that he will be directed to his intended site and tries to look for the evidence that supports his belief (which is the convincing web page) but not disproves it (which is the toolbar warning).

- Continuously showing the system model conditions the user to ignore it. Most of the time (except under phishing attacks), the site's appearance matches its identity, and the user's mental model fits the system model. In this situation, checking the system model is an extra effort to the user without any noticeable benefit.

- Even though users do notice the system model displayed by the toolbar under phishing attacks, most users do not have the expertise to correctly interpret it. For example, they cannot tell the difference between a lock icon displayed in a web page and the one displayed in the status bar. Nor are they sure whether two sites having related hostnames (e.g., amazon.com vs. amazon-department.com) are actually from the same organization in the real world. Second, sloppy but common web practices cause some users to rationalize the violation of the security rules that some indicators use to detect phishing attacks. For example, users are told to examine the hostname displayed in the address bar, to make sure that the hostname is the one they are expecting. But some legitimate websites use IP addresses instead of hostnames (e.g., the Google cache) and some sites use domain names that are totally different from their brand names. Users are also told to find the SSL lock icon before submitting sensitive information. But many legitimate banks still use unprotected login pages. [33] Such practices make it even harder for users to distinguish legitimate websites from malicious attacks. Third, Some toolbars deliver warnings without detailed convincing explanations, which makes users think that the software is buggy and not treat the warning seriously. And last, many users have no idea how sophisticated an attack could be, and do not know good practices for staying safe online.

- The security-related information shown by the toolbars is not really needed for the user's current task. Since maintaining security is rarely a user's primary goal, users fail to pay continuous attention to the toolbars. Making security a separate task that users are required to remember is not an effective solution.

- Simply warning users that something is wrong and advising them not to proceed is not a complete solution. Users will take risks to finish the tasks they think worthwhile and are not good at evaluating the tradeoffs between the claimed benefits and the potential risks.

Chapter 4

Making Security Indicators Usable

One major reason that the security toolbars do not work is that maintaining security is a separate task for the users when they are doing their primary tasks. Users cannot pay continuous attention to a separate security indicator and sometimes just ignore its warning. In this chapter, I introduce a way to integrate the security indicators into the user's main workflow so that security is not a separate task anymore, and security indicators cannot be ignored. This technique is part of a server-based cell-phone authentication solution that I have designed and implemented to prevent the user's password being captured by attackers using either active phishing or passive keyboard logging. This chapter first introduces the authentication protocol, and then presents a user study to prove this solution is both secure and user-friendly. Finally I discuss how to make the security indicator a usable and effective security measurement.

4.1 Secure web authentication using cell phones

As mentioned in Chapter 2, two-factor authentication has been used to prevent phishing attacks. One problem of two factor authentication is that users have to carry a separate authentication token, like S/KEY token or RSA SecurID, with them all the time. Moreover, some tokens require a special token reader, either hardware or software, which prevents users logging in from computers without the reader installed.

On the other hand, the cell phone has high rate of acceptability and high rate of penetration. People carry their cell phones all the time. A survey [50] showed that in 2003, 66% American adults owned a mobile phone. In some markets in Europe and Asia, mobile phone ownership is as high as 85%, and the numbers continue to grow.

Therefore, the cell phone is a good candidate as an authentication token. Moreover, the security of a two-factor authentication also depends upon the fact that the user is in possession of his cell phone. It is a reasonable assumption that when people lose their cell phones, they are typically reported lost and deactivated. Once deactivated, the registered cell phone, even in the attacker's hand, cannot be used as a valid token.

There are several solutions that use a cell phone as an authentication token. RSA Mobile [63] receives a one-time password through an SMS message. Users are required to type the receive one-time password into the browser. Fujitsu [61] proposes the same approach. Ubisecure [70] instead generates a one-time password from the cell phone itself. All these solutions share a usability problem: asking users to enter an unfamiliar one-time password at login time.

A more usable solution is proposed. Instead of typing the one-time password to login, a user only need to approve the login session from his cell phone, where the session is named by a randomly generated word.

4.1.1 Authentication protocol

In order to login to a web site that supports this authentication protocol, a user has to register his cell phone number at this site beforehand. Figure 4-1 illustrates the authentication process with seven steps:

1. The user connects his browser to the login page and types his username.

2. The username is sent to the site.

3. The site randomly chooses a word from a dictionary and displays it as a *session name* on the user's browser.

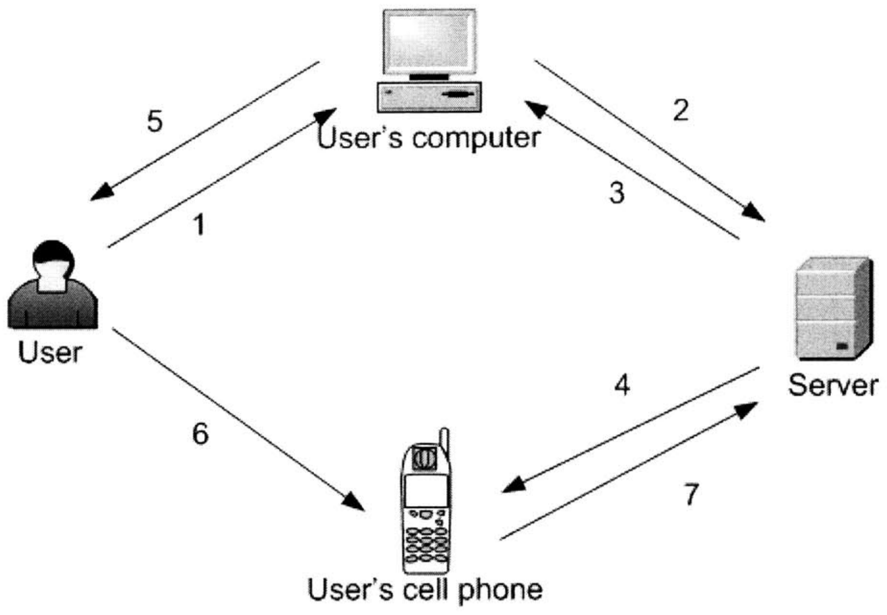


Figure 4-1: Cell-phone authentication protocol

4. The same word is sent to the user's cell phone in an SMS message. This SMS message contains a link which, once is clicked, connects the cell phone's built-in WAP browser to a dynamically generated page at the site.

5. The user looks at the session name displayed in his browser.

6. He approves the same session name through his cell phone.

7. The cell phone sends the user's approval to the site in order to authenticate the user's login session from the browser. At this point, the user is successfully logged in.

The session name displayed in the browser is an important security indicator. It represents the user's current login session. If a user approves a session name different from the one displayed in the browser, he allows a different login session with his own account. If that login session was launched by an attacker, the user's account will be compromised. Therefore, it is most important that a user should only approve a session name from his cell phone that is the same as the one displayed in the browser.

If the user chooses to disallow the session name from his cell phone, he is asked if a fraud has happened. Fraud might result if an attacker knows the user's registered username and his cell phone number at a site. (Section 4.1.3 discusses in detail how an attack can happen.) If a user sees a login request waiting for his approval at his cell phone but he does not even start a login session or the session name of the request is different from the name of his current login, he knows that he is being attacked and can decide to lock his account until further notice.

When a user finishes with the site, he can terminate his current session either by closing the browser or by revisiting the approval page on the cell phone and disconnect the session. Being able to terminate the session from the cell phone is useful for those cases when the user walks away from the computer but forgets to log out. The site will also expire the user's session after a short idle time.

4.1.2 Security analysis

By assuming that the wireless channel through the user's cell phone is secure, this solution is designed to address the following attacks:

- The *keylogger attack* that uses a hostile keylogger running in the user's computer to monitor the user's keyboard typing and capture the user's login password. The keylogger attack is a serious attack, especially at public Internet kiosks. [38, 45, 49] In this protocol, there is no password to be captured.

- The *forging attack*, in which an attacker authenticates his login session by forging an approval as an legitimate user. A random nonce is transmitted together with the SMS message sent to the user's cell phone and is sent back with the user's approval message. Without possession of the user's cell phone, an attacker cannot know the nonce and in turn forge the approval message.

- Traditional phishing attacks that send spam to arbitrary Internet users and use a phishing site to spoof the legitimate one. First, there is no password involved. Second, without knowing a user's cell phone number, the SMS message cannot be sent and the login process cannot be spoofed.

However, this solution is vulnerable to the man-in-the-middle attack, where the attacker can modify and relay the messages between the user's computer and the site. Moreover, if an attacker knows the user's registered phone number, he can launch a spear phishing attack [51] that is specific to that user by simulating the login process and then asking for other sensitive information, like credit card or bank account information.

4.1.3 User interfaces and user study

Two user interfaces

Figure 4-2(a) shows the first interface I designed. Users have to visually check the two session names, one displayed in the browser and the other displayed in the cell phone. If the two session names are same, users can approve the session. This is like the traditional "are you sure" confirmation with yes/no options. The problem with this interface is that users may not bother to check the session names and simply blindly approve the login request, which may be from an attacker. Checking the session names is a separate task from the main login task.

Figure 4-2(b) shows an improved interface. Users have to choose the correct session name that matches the browser's display from a list of randomly generated choices. This interface integrates the user's security decision into his login workflow. Without looking at the browser's display and choosing the correct name, the login cannot continue.

User study design

A user study was run to test the two interfaces and compare them, in terms of security and usability, with solutions that send a one-time password with an SMS message. The following four login techniques were studied:

- *Type random code*: A user receives an SMS message containing a one-time password consisting of eight random digits and then enters it in the browser.

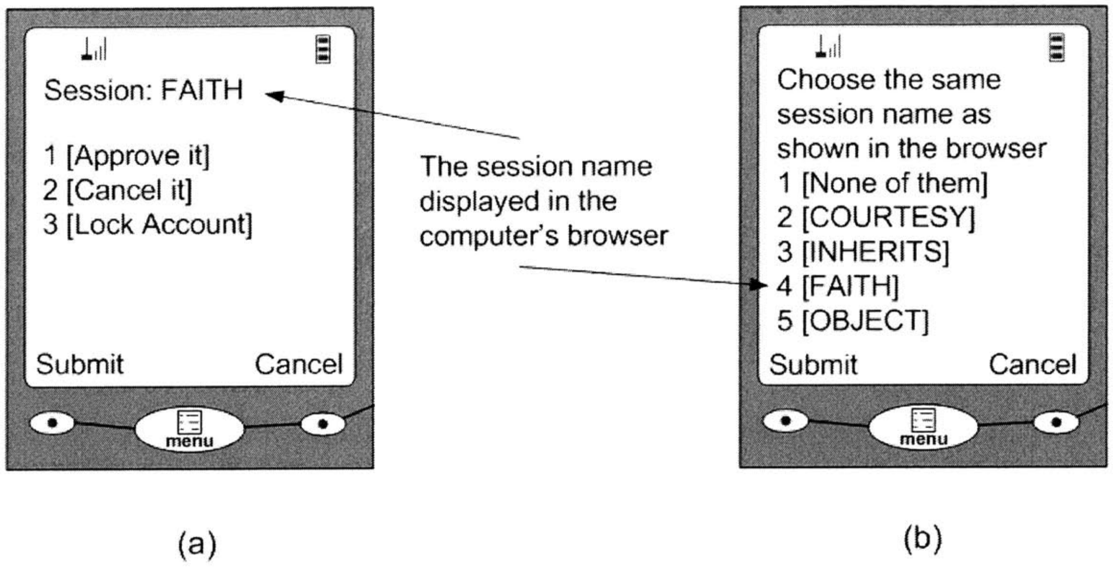


Figure 4-2: Two cell phone user interfaces

- *Type random phrase*: A user receives an SMS message containing a one-time password consisting of two random words and then enters it in the browser.

- *Check and approve* (figure 4-2(a)): A user visually checks the two session names, one displayed in the cell phone and the other displayed in the browser. If the two session names match, the user approves the current session through his cell phone.

- *Choose and approve* (figure 4-2(b)): A user chooses the correct session name that matches the one displayed in the browser from a list of choices in his cell phone.

The first two techniques simulated the one-time password approaches. To make it more user-friendly, a spelling checker was added at the server side so that when a user types two random words as the password, a mistyped non-word is corrected with a similar word, without increasing the success rate of a brute-force attack.

Simulated attacks

In theory, as shown in section 4.1.2, this solution is reasonably secure if people use them correctly. In practice, however, if a user blindly approves login sessions without looking at the session name, an attacker can trick him into approving a wrong session. Two types of attacks simulated in the study specifically target the two interfaces in figure 4-2.

The *duplicated attack* means that after a user types his username, he receives two SMS messages consecutively, one with his own session name (same as the one in his browser) and the other with the attacker's session name. The messages are sent in such an order that the user always first opens the message with the wrong session name. This attack can happen when the Internet connection from the user's computer is monitored by the attacker. Assuming that the site is SSL protected, the attacker can detect that a login connection is happening but he can not read the exact message. When the attacker detects an connection, he starts another login connection to the same site with the user's username. The site will in turn send two login requests to the user's phone.

The *blocking attack* means that after a user types his username, he receives an SMS message with the attacker's session name. At the same time, his connection with the site is blocked and there is no session name displayed in the browser. To make this attack happen, the attacker needs not only monitor but also actively control the Internet connection from the user's computer so that he can block the user's login session.

Study participants and study protocol

Twenty subjects who have owned a cell phone and have experience with Internet, 11 males and 9 females, were recruited by online and poster advertising at MIT. The average age was 25 (range from 18 to 43).

In the study, the subjects were asked to log in to a dummy site using a provided personal computer and a provided cell phone. They were first given a tutorial about the four login techniques and tried all the techniques. During the tutorial, they were free to ask questions about the login process, such as how to check the SMS message or how to use the WAP browser. They were also told that they will be attacked and were asked to protect the login account from being "stolen".

After the tutorial, each subject logged in six times in a row using each technique, for a total of 24 logins. The order of the login techniques being tested was randomized. One duplicated attack and one blocking attack were randomly assigned to either the "check and approve" login or the "choose and approve" login respectively. That is, if a duplicated attack is assigned to the "check and approve" login, then a blocking attack is assigned to the "choose and approve" login. The attack happened in the 5th or the 6th login in each row.

Results and discussion

The study results show that the "check and approve" login is easy to spoof. Four out of 11 subjects (36%) mistakenly approved the attacker's session under the duplicated attacks and two out of nine subjects (22%) approved the attacker's session under the blocking attacks. Moreover, those tricked subjects did not even have any suspicions about the attacks. They *did* notice some strange signs, but they explained the attacks away as system errors, and then approved the wrong session name. One subject tricked by the duplicated attack said, "There must be a bug in the proxy since the session name displayed in the computer does not match the one in the cell phone." Another subject tricked by the blocking attack said, "The network connection must be really slow since the session name has not been displayed yet."

The "choose and approve" login is secure with no subjects being tricked. The subjects had to pay attention to their session names in order to login. Under attacks, they couldn't find the matched choice and they either chose the "none of them" option or reported fraud.

Subjects rated "check and approve" as the easiest of the four techniques, followed by "choose and approve" (figure 4-3). Typing a one-time password was least preferred, because it forces switching between the mobile phone and the keyboard, and because

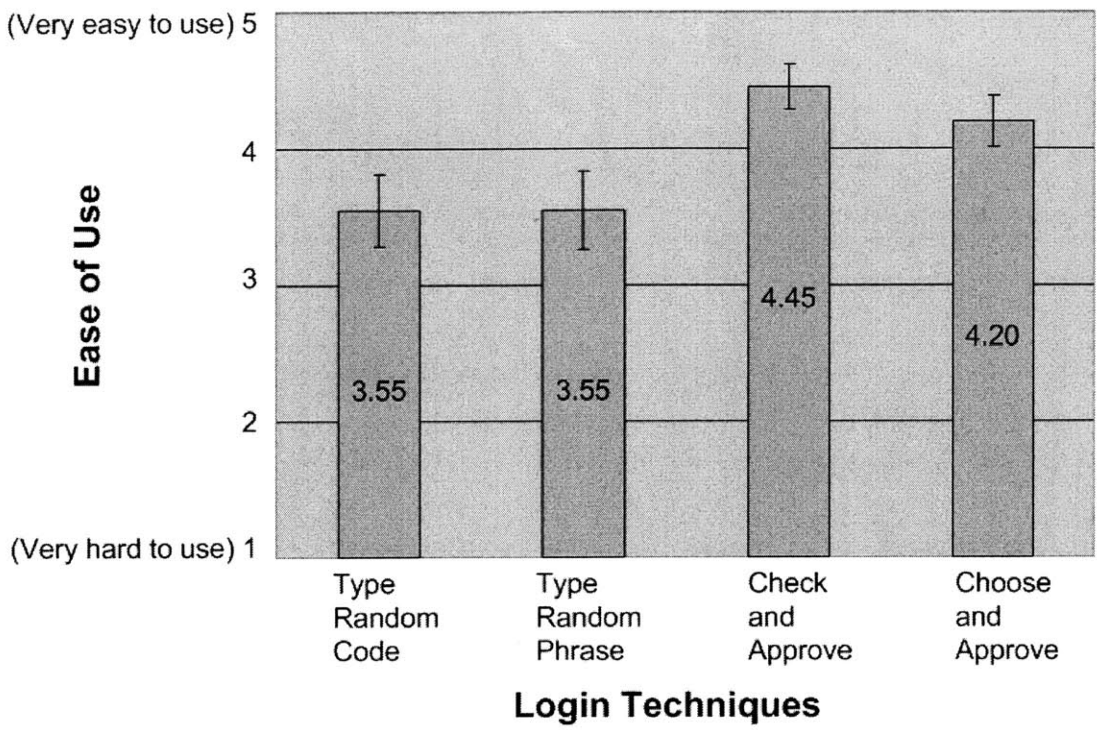


Figure 4-3: Subjective ratings on how easy to use the four login techniques

typos caused more errors. A single-factor ANOVA test shows that the differences among the subjective rates were significant with p = 0.01. “Choose and approve” login is more usable without compromising security.

The login duration was measured from the time when the subject typed the given username and clicked the "log in" button at the login page till the time when the subject's connection was successfully authenticated. The logins with attacks presented and with noticeably long SMS and WAP delays (> 20 seconds) are not included. Figure 4-4 shows the login durations. All login techniques include an SMS latency of an average of 6 seconds. "Check and Approve" and "Choose and Approve" also include a WAP latency of an average of 1.5 seconds. By a single-factor ANOVA, the differences of the login durations are significant with p = 0.02. But the time differences are not much. One reason for the “choose and approve” login having the longest duration is that the screen of the tested phone is too small to show the entire

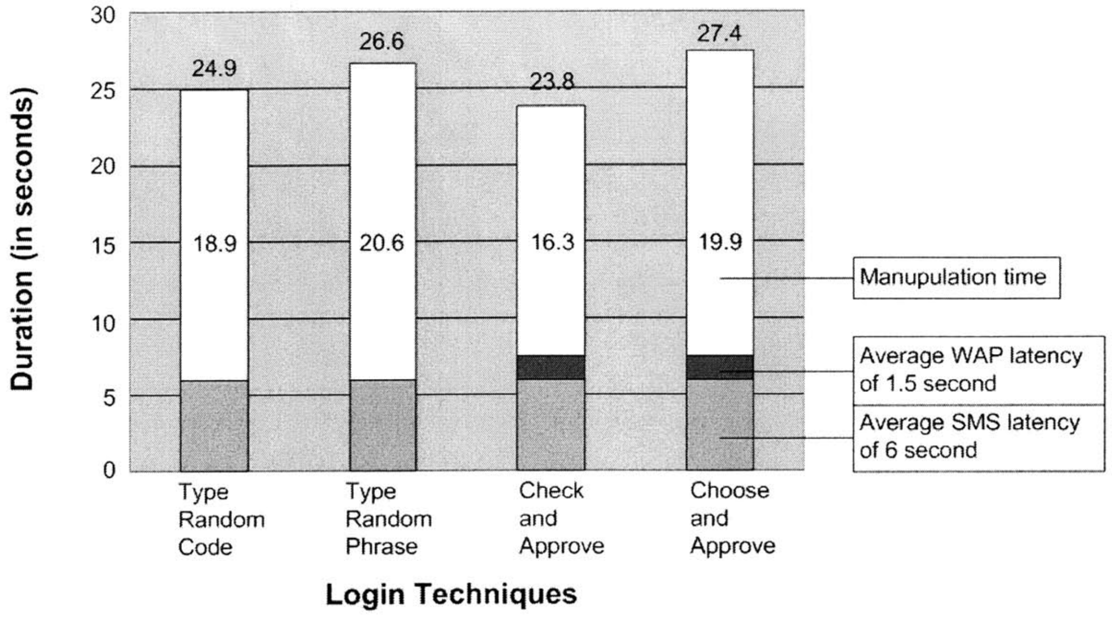


Figure 4-4: Login durations of the four login techniques

list and the subjects had to scroll over the list using the navigation buttons to find the correct session name. However, all four login techniques were much slower compared to the usual login by typing the username and the password.

4.1.4 Limitations of the cell-phone authentication as an anti-phishing solution

By asking the user to choose and approve a correct session name from his cell phone, this server-based cell-phone authentication solution is both secure and easy to use and does not require users to bring a separate authentication token. It can also be a good backup to the password login. When a user forgets his password, he can still log in using his cell phone instead.

But this solution is not a complete anti-phishing solution, because of the following limitations. As mentioned in section 4.1.2, it cannot prevent the man-in-the-middle attack, and it cannot protect other types of sensitive information, only passwords. Moreover, it requires server-side changes to support the authentication protocol and cannot protect servers that do not support it.

4.2 How to make security indicators usable?

Both the toolbar studies and the cell-phone authentication study show that users fail to continuously check the security indicators if it is a separate task, especially when most of the time the security indicator means nothing to them (i.e., the indicator only helps when an attack happens). The "choose and approve" approach shows that moving the security indicator to the user's main workflow can better protect users. It provides a way to improve the traditional "are you sure" confirmation mechanism.

"Choose and approve" can used as an anti-phishing method. Under phishing attacks, users think that their data is submitted to one site but the data is actually sent to another different site. Toolbars that show the site information are not effective enough to help users to detect the discrepancy. Instead, the users can be asked to explicitly choose the site where they want the data to go from a list. Under phishing attacks, if the list can include the spoofed legitimate site, which is also the site that the users want their data to go, the users are likely to choose it and as a result the discrepancy is detected.

However, as shown in figure 4-3 and figure 4-4, "choose and approve" is less usable than "check and approve". It sacrifices usability to security. Users have to choose an option from a list and sometimes the list may not be short. Therefore, "Choose and approve" is not an universal replacement to the traditional "are you sure" confirmation. It has to be used carefully in order to minimize the side effect to usability. For example, users could be asked to choose their intended site only when they are submitting sensitive information to a site and it is their first time to submit this information to this site. I will discuss in detail in the next chapter how the "choose and approve" approach can be used in a new anti-phishing solution called Web Wallet.

Chapter 5

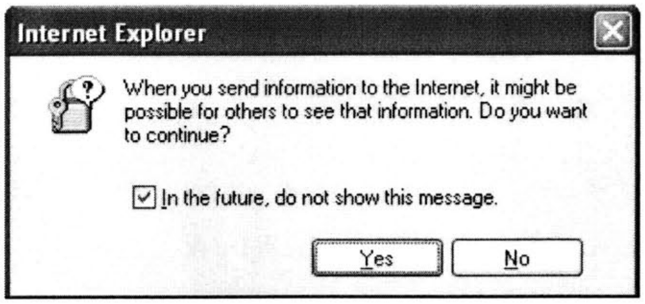
Web Wallet User Interface and its Design Principles

This chapter introduces Web Wallet, a new anti-phishing solution. Web Wallet addresses several serious problems that make online interaction vulnerable to phishing and make anti-phishing toolbars fail to prevent high-quality phishing attacks.

Using a web form to submit sensitive information is a common practice on legitimate sites. But it has two problems that make phishing attacks easier.

- First, the appearance of a web site and its web forms are easy to spoof. A web site can control what it looks like in a user's browser, so a site's appearance does not reliably reflect the site's true identity. But users tend to decide site identity based on appearance, e.g., "This site looks exactly like the PayPal site that I have been to before. So it must be a PayPal site." As a result, users may be tricked into submitting data to phishing sites.

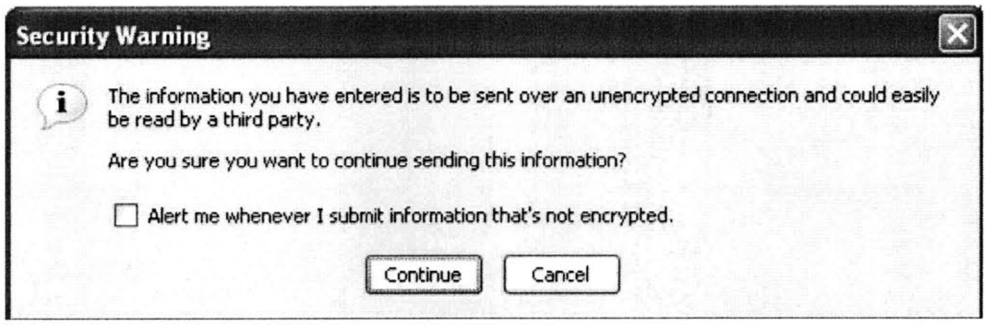
- Second, web forms are used for submitting insensitive data as well as sensitive data. Even though SSL encryption can indicate to the browser that the input data is sensitive, phishing sites rarely use SSL and the browser fails to effectively visually differentiate an SSL connection from a non-SSL one. Moreover, the semantic meaning of the input data is opaque to the browser. To the browser, a credit card number used to buy a book and a search query for the name of a


Internet Explorer

When you send information to the Internet, it might be possible for others to see that information. Do you want to continue?

[✓] In the future, do not show this message.

Data submission warning from Internet Explorer 6


Security Warning

The information you have entered is to be sent over an unencrypted connection and could easily be read by a third party.

Are you sure you want to continue sending this information?

[ ] Alert me whenever I submit information that's not encrypted.

Data submission warning from Firefox 1.5

Figure 5-1: Browser warnings about data submission

book are just two strings of bits. Therefore, the browser fails to give appropriate protections to data submissions with different sensitivities. With the default configuration, a browser treats every data submission as sensitive and pops up a warning (figure 5-1) if the data submission is not SSL protected. However, with the warning dialog, users can tell the browser not to show the warning anymore. The browser will then treat every data submission as insensitive. This kind of "all-or-none" warning mechanism fails to differentiate data submissions based on their sensitivity.

These two problems show that under phishing attacks, there is a semantic gap between a user's mental model (e.g., "submit my credit card information at Amazon.com") and the system model (e.g., "post a string of bits to the server with the IP address of 69.10.142.34") of an online interaction. This is the fundamental danger of phishing, and in order to prevent phishing attacks, this gap has to be bridged.

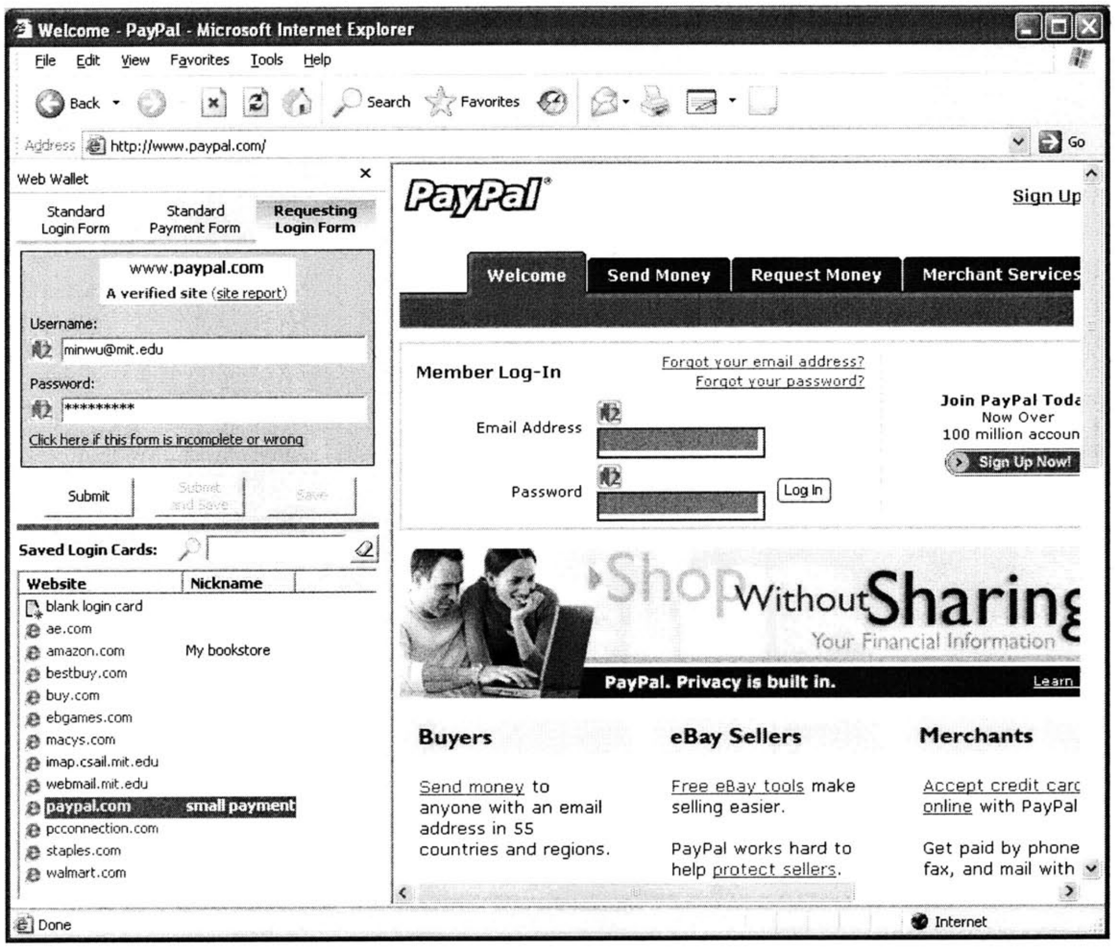
Existing approaches try to bridge this gap by showing the system model to the user. For example, the anti-phishing toolbars display various security messages to help users detect phishing sites. These messages are mainly about system properties of the web site, such as whether the current page is protected by SSL, when and where this site was registered, or whether this site is in a blacklist of fraudulent sites. But as section 3.7 shows, these toolbars are not effective.

Unlike the anti-phishing toolbars that show the system model to the user, Web Wallet takes the reverse approach, by reflecting the user's real intention to the system. The main part of Web Wallet is a browser sidebar for entering sensitive information (figure 5-2). When a user sees a web form requesting sensitive data, he presses a dedicated security key on the keyboard to open Web Wallet, telling the system that he is going to submit sensitive data and the system should take extra care about this submission. Using Web Wallet, he may type his data or retrieve the sensitive data that he has stored into Web Wallet before. The data is then filled into the web form. But before the fill-in, Web Wallet checks if the current site is good enough to receive the sensitive data. If the current site is not qualified, Web Wallet requires the user to explicitly indicate where he wants the data to go. If the user's intended site is not the current site (which probably indicates phishing), Web Wallet shows a warning to the user about this discrepancy, and gives him a safe path to his intended site.

5.1 Web Wallet's user interface design principles

Web Wallet is based on three design principles:

- An *intention-centered* design approach that makes the user's intention obvious to the system;
- Integrating security into the user's task workflow so that the security mechanism cannot be ignored;



www.paypal.com
A verified site (site report)

Username: minwu@mit.edu

Password: *********

Click here if this form is incomplete or wrong

Saved Login Cards:

Website	Nickname
blank login card	
ae.com	
amazon.com	My bookstore
bestbuy.com	
buy.com	
ebgames.com	
macys.com	
imap.csail.mit.edu	
webmail.mit.edu	
paypal.com	small payment
pcconnection.com	
staples.com	
walmart.com	

Member Log-In
Forgot your email address?
Forgot your password?
Email Address
Password

Join PayPal Toda
Now Over
100 million accoun

Buyers
Send money to anyone with an email address in 55 countries and regions.

eBay Sellers
Free eBay tools make selling easier.
PayPal works hard to help protect sellers.

Merchants
Accept credit carc online with PayPal
Get paid by phone fax, and mail with

Figure 5-2: Web Wallet as an Internet Explorer sidebar

- Making the safest path the path of least resistance, so that the user is most likely to choose it.

5.1.1 Intention-centered design

When a user is submitting data, his intention includes two parts. The first part is the data type. Is the submitted data sensitive or not? If yes, what kind of sensitive data is it? The second part of his intention is the data recipient: which site does the user intend to submit his data to? When a user uses Web Wallet – a dedicated interface for sensitive information submission – he implicitly indicates that the submitting data is sensitive and thus shows the first part of his intention (the data type) to the system.

Using different actions to signal different intentions has been proposed by Yee to prevent email viruses. [77] On Microsoft Windows and Mac OS systems, double-clicking means either opening documents or launching applications. Users sometimes double click an email attachment, intending only to read it but instead starting an email virus. Yee suggested using a dedicated Applications folder to solve this problem. Double-clicking files in the Applications folder launches them, and double-clicking files elsewhere only passively views them. The Application folder can only contain files that the user has placed there. As a result, the system can differentiate the user's intentions between viewing and executing.

Users use Web Wallet's dedicated interface to submit sensitive data, such as passwords for logging in and credit card information for online shopping, and use traditional web forms for other submissions, like queries for search. The submission through Web Wallet needs extra care. The second part of the user's intention – the site where he intends to submit the sensitive data – should be confirmed. If there is a discrepancy between the user's intended site and the current site, Web Wallet can conclude that it is likely that the user is under a phishing attack and effectively warn him about the potential danger by a message like "you may think that this site is PayPal but in reality this site has no relationship with PayPal (here is why) and thus this site is probably fraudulent." This warning based on the mismatch between the user's intention and the system operation, directly addresses the fundamental danger of phishing. Moreover it is easier to understand and less likely to be rationalized by the user than a warning based purely on the system model, like "this site has just been registered and the current submission is not SSL protected".

Knowing the user's intended site, the system could not only effectively warn the user but also provide an alternative *safe path* to his intended site. Providing a safe path is a useful feature because the toolbar studies found that some users tend to take risks to finish the tasks they think worthwhile even they are warned about possible phishing attacks. With the safe path, they do not have to take the risk because the safe path links to the real site that they want.

The potential problem with the safe path approach is that even following the safe path, the user may not finish his current goal because most phishing attacks use fake requests, like confirming account information to avoid suspension, or claiming a prize. Some users may go back to the phishing message after they follow the safe path to their intended site and realize that their goal can not be fulfilled. A possible solution is education. As long as the user follows the safe path, it is highly probable that he is under a phishing attack. It is an appropriate time to let him know about phishing with some typical concrete examples.

5.1.2 Integrate security into the workflow

When users are doing tasks online, security is rarely their primary goal. Putting security as a separate task has been shown to fail by both the toolbar studies in chapter 3 and the cell-phone authentication study in chapter 4. Effective security mechanisms should integrate themselves into the user's workflow.

When Web Wallet is used for sensitive data submission, sensitive input fields in web forms are disabled to make Web Wallet the only way to input sensitive data. Web Wallet trains users to depend on it using most legitimate sites whose sensitive inputs can be successfully blocked.

Web Wallet incorporates security questions by helping users achieve their goals instead of stopping them. When Web Wallet confirms the user's intended site, it does not use a generic warning like "are you sure you want to send this information to this site", which is known to be ineffective because a user tends to say yes – meaning not that the current site is his intended site but that he definitely wants to continue making progress towards his goal. [53] Instead, users are shown a list of sites, including the current site, and are required to explicitly acknowledge and indicate their intended site. The cell-phone authentication user study showed that asking a user to actively choose a right way to finish his task is more reliable than asking the user to passively confirm that he is in a right way to finish his task.

5.1.3 The safest path should be the path of least resistance

The path of least resistance is the most natural and easiest way for a user to do his task. Ideally, all the sensitive input fields in the web forms are disabled so that users have to use Web Wallet to submit sensitive data. But it should be also acknowledged that users are accustomed to web form submission and have a strong tendency to use it, even though it is an insecure practice that is vulnerable to phishing. Web Wallet can disable the sensitive forms from most legitimate sites to train users in the new habit of using it. But if it is possible that a phishing site can present an enabled sensitive form to the user, users may fall back to their old habit to use the form. It is important to make Web Wallet's submission mechanism always available and ready for use. Any difficulties in using Web Wallet may make users switch to the enabled web forms under phishing attacks.

Another way to make the Web Wallet submission the most natural way is to make the web form submission awkward to use, especially when submitting sensitive data, by using disturbing visual cues indicating the web form submission is in general not a safe practice.

There are two types of security-related visual cues. A positive cue is used to indicate a safe mode with its appearance. A typical example of the positive cue is the SSL icon in the browser's status bar, whose appearance indicates that the current page is SSL protected. The absence of the positive cue means a unsafe mode.

A negative cue, on the other hand, is used to indicate a unsafe mode with its appearance. An example of a negative cue is the warning message from the Trustbar (figure 2-5) saying the current page is not protected by SSL. Moreover, unlike the popup warnings shown in figure 5-1, a negative cue does not block the user's workflow in order to get the user's attention. Because it is harder for humans to detect the absence of something, as opposed to its presence [68], the presence of a negative cue, if used correctly, is expected to be more effective than the absence of a positive cue in order to indicate a unsafe mode. One problem of the Trustbar warning is that it is displayed in a peripheral area, not in the center of the user's attention.

5.2 Web Wallet user interface

Currently Web Wallet supports submission of login information and credit card information. This section introduces the Web Wallet user interface by going through the submission process.

5.2.1 Sensitive input annotation

For every web page that is displayed in the user's browser, Web Wallet searches for HTML input elements that ask for either login information or credit card information. When sensitive inputs are detected, they are disabled so that they no longer permit typing. A web wallet icon is inserted before and a green boundary is added around each sensitive input to indicate that Web Wallet should be used. Related inputs are grouped into a single protected area so that they can be filled together from Web Wallet (figure 5-3).

When there are several protected areas in a single page, the first area is by default active (figure 5-3). That is, when Web Wallet is dealing with the data submission for that area. Users can activate a different protected area by clicking one of its elements on the page.

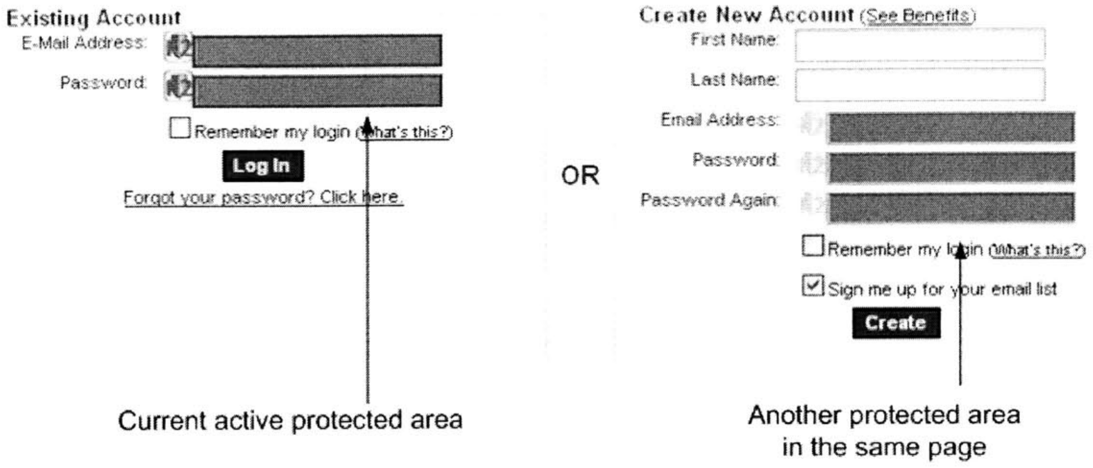
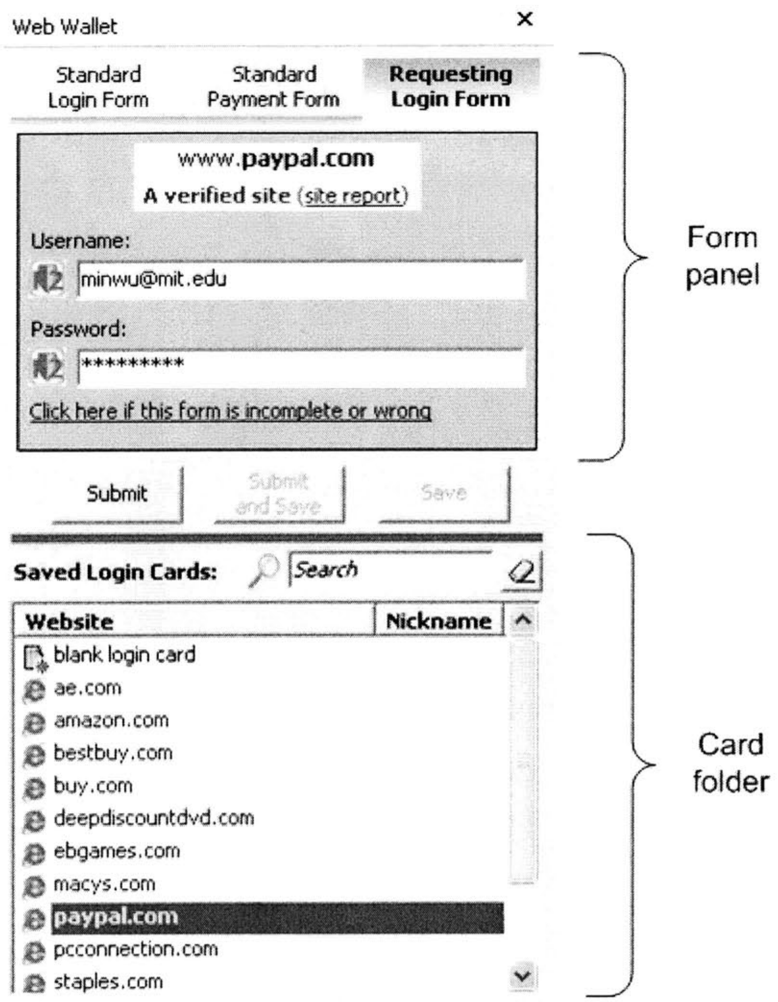

Existing Account
E-Mail Address:
Password:
Remember my login (What's this?)
Log In
Forgot your password? Click here.
OR
Create New Account (See Benefits)
First Name:
Last Name:
Email Address:
Password:
Password Again:
Remember my login (What's this?)
Sign me up for your email list
Create
Current active protected area
Another protected area in the same page

Figure 5-3: Web Wallet annotates and disables sensitive inputs on web pages

5.2.2 Security key

Even if there are detected protected areas in the web page, Web Wallet does not open automatically; rather, the user is required to press the security key to open it, for two reasons. First, the user may not want to submit data. Many web sites include a login form in their home page, and the user may simply want to browse the site with no intention of logging in. Second, pressing the security key needs to be an essential action so that it becomes habitual. When the user forms this habit, he will always open the real Web Wallet every time he wants to use it.

In the current implementation, F2 is used as the security key. The F2 key has been proposed as a security key [57] because it is not currently used by all major browsers. However, F2 by itself does not have any security implication. It is better to have a dedicated security key in the keyboard. The security key idea has been implemented for a long time by the Windows system to bring the user to an authentic login screen. Pressing a security key to open a trusted interface is more secure than clicking somewhere on the screen because every target on the screen can potentially be spoofed. The event of pressing the security key is first handled by Web Wallet and then is discarded, so that its activation cannot be hijacked or even detected by a phishing web page.



www.paypal.com

A verified site (site report)

Username: minwu@mit.edu

Password: *********

Website	Nickname
blank login card	
ae.com	
amazon.com	
bestbuy.com	
buy.com	
deepdiscountdvd.com	
ebgames.com	
macys.com	
paypal.com	
pcconnection.com	
staples.com	

Figure 5-4: Web Wallet interface

5.2.3 Browser sidebar

When the user presses F2, Web Wallet is opened as a browser sidebar (figure 5-4). The main interface contains two parts: a form panel and a card folder.

Site reputation display

The site hostname and a summary of the site's reputation information are displayed in the form panel, and the background color of the panel reflects the site reputation (figure 5-5). Internet sites are separated into four categories by Web Wallet. A *verified site* with a green background, means this site has been verified by some trusted authority. The site is expected to be securely maintained and the user's interaction with this site is safe. A *known fraudulent site* with a red background, means this site has been detected and reported as fraudulent. Sensitive data submission to this site is highly discouraged.

There still are a lot of sites in the gray area between verified (as a whitelist) and known fraudulent (as a blacklist). These sites are separated into two classes based on the *longevity* of the site. A common feature of phishing web sites that discriminates them from the web sites they are imitating is their short duration. Phishing sites may exist for only days or hours before the attackers take it down or authorities have it removed. Thus the longevity of a site is a critical part of its reputation. Estimates of longevity can be obtained by checking the Internet Archive, counting incoming links from other sites, or consulting WHOIS database records to learn when the domain was registered. Sites with no registration information and not linked by any other sites are rated as *suspicious*, with a red background. All other sites, with some proof about their age are *unverified* sites with a yellow background.

Detailed site reputation information can be shown in a popup window when a user clicks the "site report" link (figure 5-6). The following four pieces of information, if available, are shown to help users better recognize the current site.

- *SSL protection*: A site is protected by SSL is a good trust indicator. Most legitimate web sites use SSL to protect sensitive information submission. On the other hand, few phishing sites use SSL because obtaining an SSL certificate from a well-known Certificate Authority (CA) requires site identity information that can be traced. But SSL should not be the only factor used in the trust analysis because some phishing attacks do use SSL, thanks to sloppy practices by some CAs. [42]

- *Trusted third-party certificate*: Legitimate web sites display the certificates that they get from the trusted authorities, like TRUSTe and BBBOnLine, to indicate that they are trustworthy. These certificates are another good trust indicator.

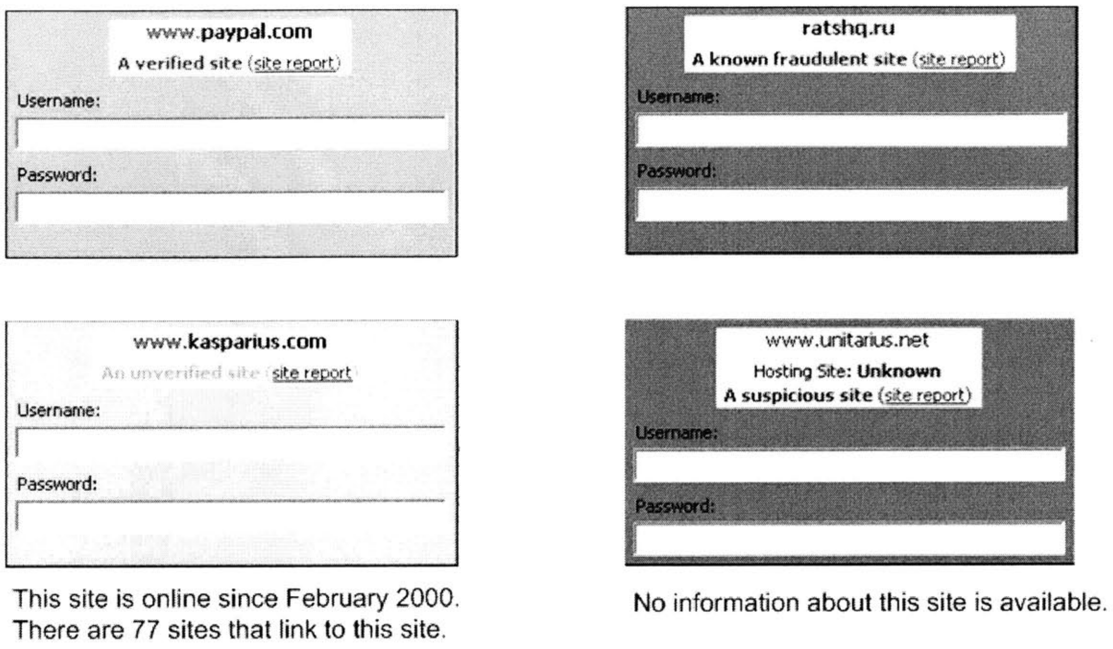

www.paypal.com
A verified site (site report)
Username:
Password:
ratshq.ru
A known fraudulent site (site report)
Username:
Password:
www.kasparius.com
An unverified site (site report)
Username:
Password:
This site is online since February 2000.
There are 77 sites that link to this site.
www.unitarius.net
Hosting Site: Unknown
A suspicious site (site report)
Username:
Password:
No information about this site is available.

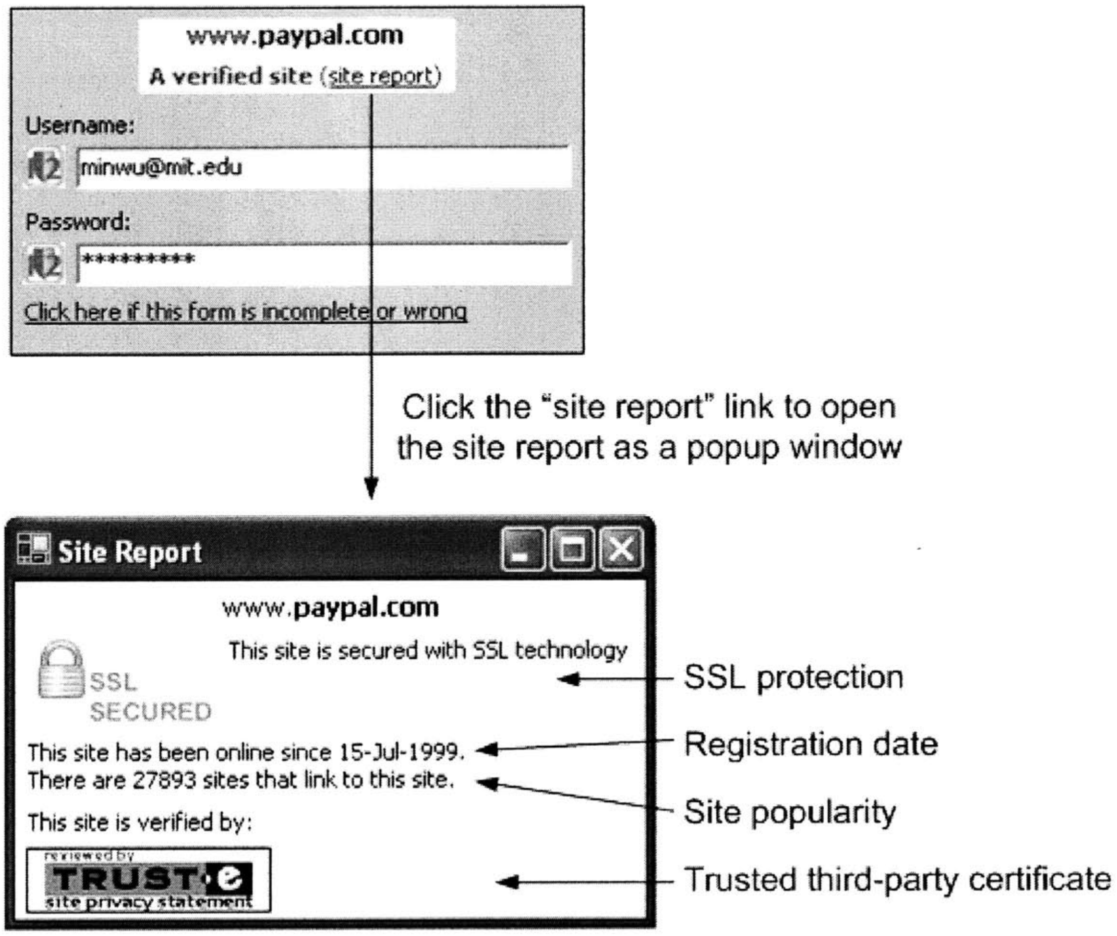
Figure 5-5: Different form panel displayed based on the site reputation

- *Site popularity*: Site popularity is measured by how many other sites link to this site. Legitimate sites want to be popular in order to attract more users, but phishing sites do not want to be popular in order to avoid public inspection. Therefore, the site popularity is a useful indicator to include in the site report. It may help users detect a phishing site if they notice that a site claims to be a well known site but actually there are no other sites that link to it.

- *Site registration date*: Given the fact that most phishing sites are short-lived, the site's registration date is a useful indicator to include in the site report. It may help users detect a phishing site if they notice that a site claims to be a well-known site but it was actually registered just a few days ago.

Form panel

Each protected HTML element in the active protected area is mapped to an input in the form panel, which is a safe place for users to input their sensitive data. On the


www.paypal.com

A verified site (site report)

Username:

minwu@mit.edu

Password:

****

Click here if this form is incomplete or wrong

Site Report

www.paypal.com

This site is secured with SSL technology

SSL SECURED

This site has been online since 15-Jul-1999.

There are 27893 sites that link to this site.

This site is verified by:

reviewed by TRUSTe site privacy statement

Click the "site report" link to open the site report as a popup window
SSL protection
Registration date
Site popularity
Trusted third-party certificate

Figure 5-6: Site report as a popup window

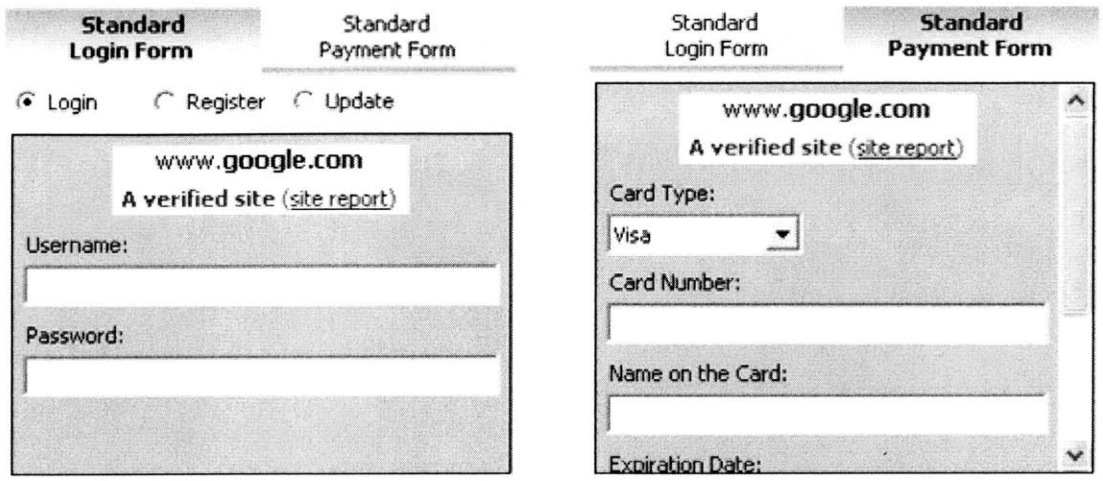


Figure 5-7: Standard login form and payment form

other hand, when Web Wallet is open but there is no protected area in the page, the form panel displays a standard login form and a standard payment form (figure 5-7). These two standard forms allow Web Wallet to be used even when sensitive inputs fail to be detected. These two forms help to prevent phishing attacks by making Web Wallet an easiest submission method, as will be discussed in chapter 7. Moreover, with the two standard forms, users can modify their stored cards at Web Wallet.

Chapter 6 will discuss in detail how Web Wallet uses machine learning algorithms and a set of heuristic semantic rules to detect sensitive inputs in web pages and group them into protected areas. Because legitimate web sites tend to request login information and credit card information in a consistent way, i.e., using the same terminology for sensitive information and asking for different types of sensitive information in a same order (first username then password), the detection algorithm has high accuracy. But the sensitive-input detection algorithm does have errors. For example, it sometimes fails to detect some sensitive inputs and sometimes mistakenly detect some insensitive inputs as sensitive.

There are five common detection errors:

1. A sensitive input that should be included in a protected area fails to be included (figure 5-8(a)).

2. An insensitive input that should not be included in a protected area is mistakenly included (figure 5-8(b)).

3. A sensitive input is correctly included in a protected area but is assigned with the wrong sensitive type (figure 5-8(c)).

4. A protected area is mistakenly detected but all its inputs are not sensitive (figure 5-8(d)).

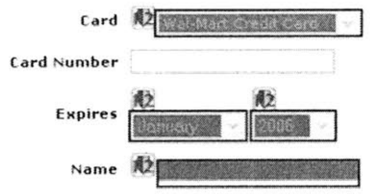
5. A group of sensitive inputs that should form a protected area totally fails to be detected (figure 5-8(e)).

The form panel, which displays the active protected area in the web page, provides a way for users to manually correct the detection errors. The user's modifications are remembered so that when the same form is displayed again, its sensitive inputs will be correctly annotated and disabled.

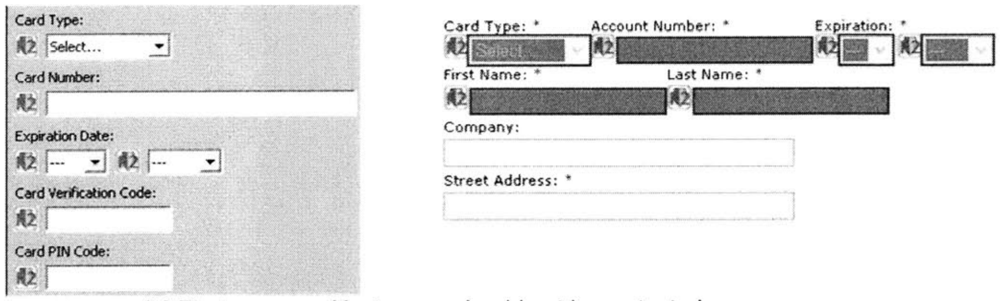
Because the detection algorithm in general has high accuracy, the manual modification is hidden as advanced features by default, behind the link "Click here if this form is incomplete or wrong". When the link is clicked, users can add more sensitive types in the current protected area (dealing with detection error 1). Then when users click a submit button, while the data in the detected inputs are automatically filled into the corresponding HTML elements in the page, a drag-and-drop icon is displayed near the manually-added inputs and users can drag the icon and drop it to a specific HTML element in the page to not only fill the element but also specify the sensitive type of that element. Figure 5-9 shows how users can specify an undetected sensitive input.

Figure 5-10 shows that users can right-click the Web Wallet icon (either in the form panel or in the web page) to unprotect either the corresponding entry (dealing with detection error 2) or the whole protected area (dealing with detection error 3). Moreover, they can also change the sensitive type of the corresponding entry (dealing with detection error 4).

Users can use the two standard forms (figure 5-7) to deal with an undetected

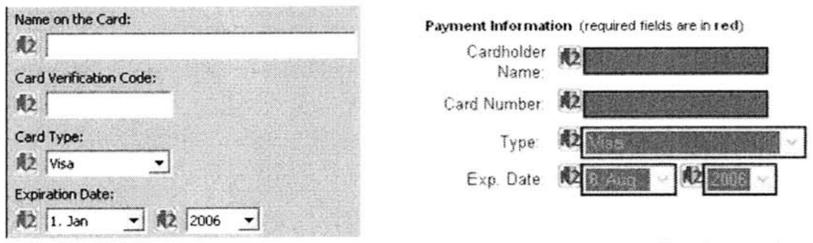

Card Wal-Mart Credit Card
Card Number
Expires January 2006
Name

(a) Credit card number fail to be protected


Card Type: Select...
Card Number:
Expiration Date: --- ---
Card Verification Code:
Card PIN Code:

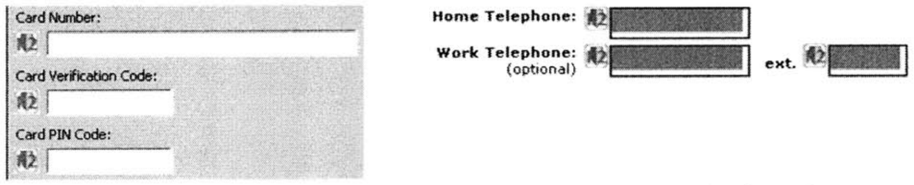
Card Type: * Select...
Account Number: *
Expiration: * -- --
First Name: *
Last Name: *
Company:
Street Address: *

(b) First name and last name should not be protected


Name on the Card:
Card Verification Code:
Card Type: Visa
Expiration Date: 1. Jan 2006

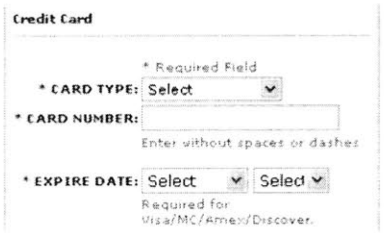
Payment Information (required fields are in red)
Cardholder Name:
Card Number:
Type: Visa
Exp. Date: 8. Aug 2006

(c) Credit card number is mistakenly detected as card verification code


Card Number:
Card Verification Code:
Card PIN Code:

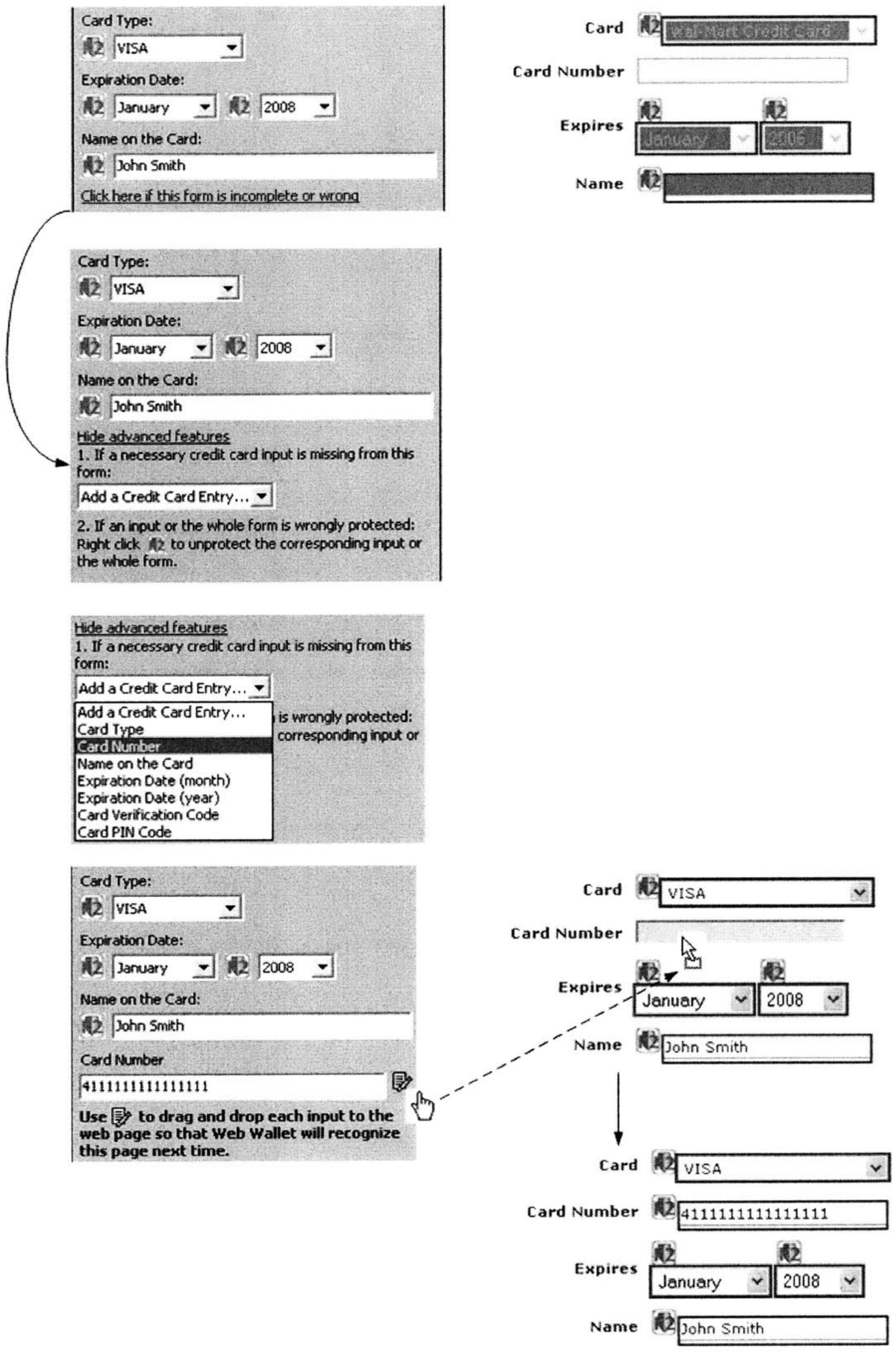
Home Telephone:
Work Telephone: (optional)
ext.

(d) Inputs for phone numbers are mistakenly protected as credit card information


Credit Card
* Required Field
* CARD TYPE: Select
* CARD NUMBER:
Enter without spaces or dashes
* EXPIRE DATE: Select Select
Required for Visa/MC/Amex/Discover.

(e) Inputs for credit card information fail to be protected

Figure 5-8: Web Wallet sensitive-input detection errors (Note: All the errors are introduced artificially only for demonstration)



Card Type: VISA

Expiration Date: January 2008

Name on the Card: John Smith

Click here if this form is incomplete or wrong

Card: Wal-Mart Credit Card

Card Number:

Expires: January 2006

Name:

Card Type: VISA

Expiration Date: January 2008

Name on the Card: John Smith

Hide advanced features

1. If a necessary credit card input is missing from this form:

Add a Credit Card Entry...

2. If an input or the whole form is wrongly protected: Right click to unprotect the corresponding input or the whole form.

Hide advanced features

1. If a necessary credit card input is missing from this form:

Add a Credit Card Entry...

- Add a Credit Card Entry...
- Card Type
- Card Number
- Name on the Card
- Expiration Date (month)
- Expiration Date (year)
- Card Verification Code
- Card PIN Code

Card Type: VISA

Expiration Date: January 2008

Name on the Card: John Smith

Card Number: 4111111111111111

Use to drag and drop each input to the web page so that Web Wallet will recognize this page next time.

Card: VISA

Card Number:

Expires: January 2008

Name: John Smith

Card: VISA

Card Number: 4111111111111111

Expires: January 2008

Name: John Smith

Figure 5-9: Web Wallet uses the drag-and-drop mechanism for users to specify undetected sensitive inputs

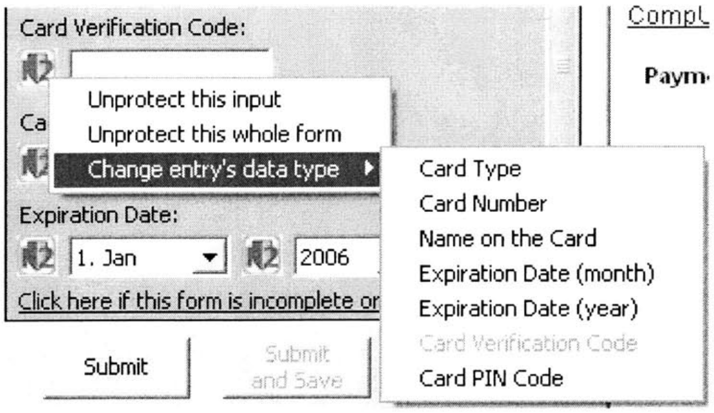


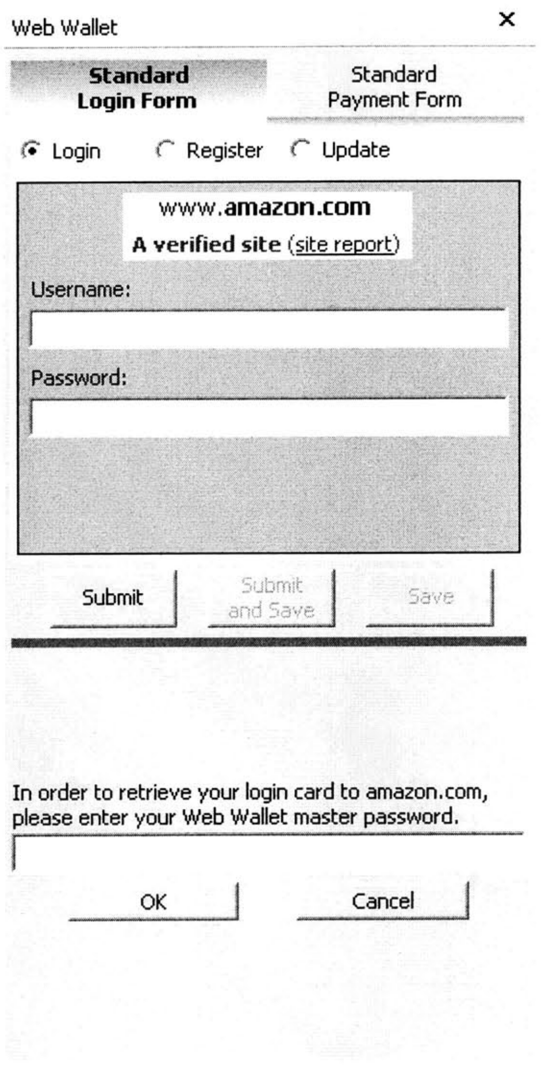
Figure 5-10: Right click the Web Wallet icon to modify the detected sensitive inputs

protected area (detection error 5). When users click a submit button, a drag-and-drop icon is displayed with each input entry and users can drag each icon one by one and drop it to an individual HTML element. All the HTML elements dragged-and-dropped from the same standard form are grouped as a single protected area.

Card folder

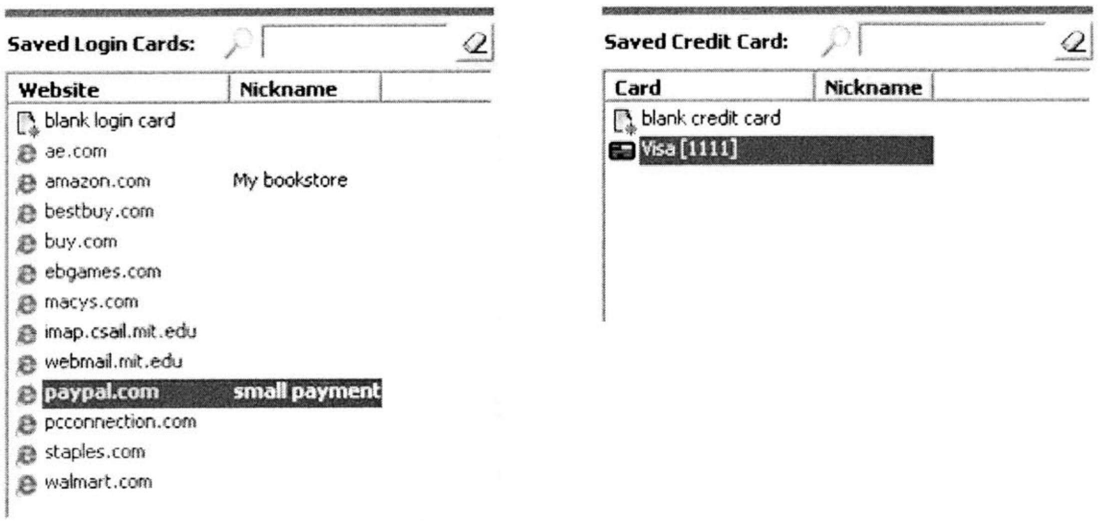
The card folder shows stored login or credit cards. The card metaphor is a natural data unit for login and credit card information, as well as other sensitive information like bank account information. The stored cards are encrypted using Web Wallet's master password. Users will not be prompted for the master password until they first interact with the card folder, i.e., when they are retrieving a stored card or saving a new card (figure 5-11).

As shown in figure 5-12, the stored login cards are indexed by the domain names of the web sites where they are used to log in. If a single domain contains several web sites that ask for different login accounts, the hostnames of the web sites are used to index the cards instead. For example, webmail.mit.edu is a web-based email service for MIT and imap.csail.mit.edu is a web-based email service for MIT's Computer Science and Artificial Intelligence Lab. Their login cards share a single domain and their indexes are shown differently, using the hostnames. Users can assign nicknames



In order to retrieve your login card to amazon.com, please enter your Web Wallet master password.

Figure 5-11: Web Wallet requests its master password when the card folder is first accessed



Saved Login Cards:

Website	Nickname
blank login card	
ae.com	
amazon.com	My bookstore
bestbuy.com	
buy.com	
ebgames.com	
macys.com	
imap.csail.mit.edu	
webmail.mit.edu	
paypal.com	small payment
pcconnection.com	
staples.com	
walmart.com	

Saved Credit Card:

Card	Nickname
blank credit card	
Visa [1111]	

Figure 5-12: Saved login cards and credit card in Web Wallet

to individual cards. The login cards can be sorted by their domain names or their nicknames. Users can also incrementally search their login cards using the domain name and the nickname. Users can have several login cards for a single web site. In this case, individual cards are differentiated by their specific login usernames.

The stored credit cards are indexed by the card type and the last four digits of the card number. If the card type is not available, the default type "Card" is used as the type.

A stored login card also acts as a bookmark. Users can right-click a login card and select the "go to site" menu to go to the corresponding web site (figure 5-13). Currently, the login card only directs users to the site's home page. It is more user-friendly if clicking the login card can automatically login the user using the stored account information, which will be discussed in section 9.2.1.

Data submission

When there is a stored card that matches the request from the active protected area, which happens when the user has sent the same card to this site before and has saved that card into Web Wallet, then the stored card data is filled into the form panel and

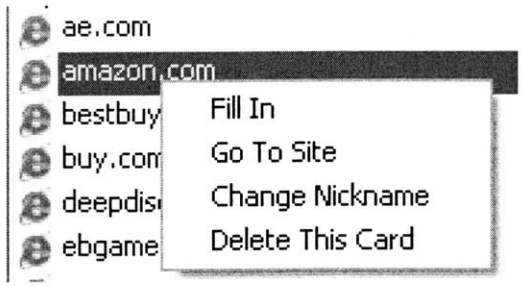


Figure 5-13: A saved login card act as a bookmark

then into the protected area. The user then clicks the submit button in the form. This procedure is free from phishing because the user has submitted the same data to this site before.

If there are several stored cards that match the request and can be filled, all the matched card in the card folder are displayed in bold face, and the matched card with the latest access time is selected by default and its data is filled into the form panel.

When there is no stored card matching the request, users have to manually fill the form panel. Users can then click the "submit" button to fill the typed data into the web form. Users click the "submit and save" button if they also want to save the typed data for later use. Saving the sensitive information at the same time when it is submitted makes Web Wallet easy to use as a personal information manager. Users do not have to store their sensitive information separately before using it, which is a necessary step required by many personal information managers.

Web Wallet does not greatly complicate a user's interaction with legitimate sites, and can even simplify the interaction. For example, when a user logs in without a stored login card, he must perform several extra steps compared with logging in directly: he must press F2, shift his attention to Web Wallet, enter his username and password as usual, press the submit button in Web Wallet, and finally shift his attention back to the web form. However, when a user has a login card stored, he only needs to press F2 to activate the auto-fill feature of Web Wallet.

5.2.4 Site confirmation dialog

When users submit their typed data from the sidebar to the web form, three cases may happen depending on the reputation of the current site and the user's history of using Web Wallet.

If the current site is a verified site, the data is submitted. If the current site is not verified but Web Wallet remembers that the user has submitted the same data to this site before, the data is also submitted. In the last case, if the site is not verified and the user has not submitted the exact data before, he will see a site confirmation dialog (figure 5-15(a)) asking him to indicate his intended site.

These three cases represent a tradeoff between usability and security. Letting users indicate their intended site is an extra step in the Web Wallet submission process. It sacrifices usability for security. By reasonably assuming that a verified site is securely maintained and a user's interaction with it is always safe, Web Wallet skips the site confirmation step at the verified sites in order to improve usability. Of course the limitation of this decision is that if a verified site *is* successfully attacked, it may launch very effective attacks to the Web Wallet user.

If a site is not verified, the system can not simply block the submission. The site may be a new site that Web Wallet does not know about. Moreover, because the user's login information is site-specific, users can register and use low-value accounts at unverified sites as long as those sites do not trick them submitting their high-value important accounts, like their bank account.

The user's submission history is used to improve usability without compromising security. If a piece of data has submitted to a site before, and thus is already known to the site, the same data can be safely re-submitted. The hash value of the submitted data indexed by the site's hostname is stored in Web Wallet. The original data cannot be recovered from the hash value. Only storing the type of the submitted data is not enough. A site may first let users register an account with it and later spoof another legitimate site and ask user's password for that targeted site.

The confirmation dialog has a site list on the left. The list is generated from the user's Web Wallet history plus the current site name. The site names in the list are sorted in alphabetic order. The reputation summary is displayed together with each site name. The user's history is used to generate this list because many effective phishing attacks claim to be legitimate sites that the user has contacted online before.

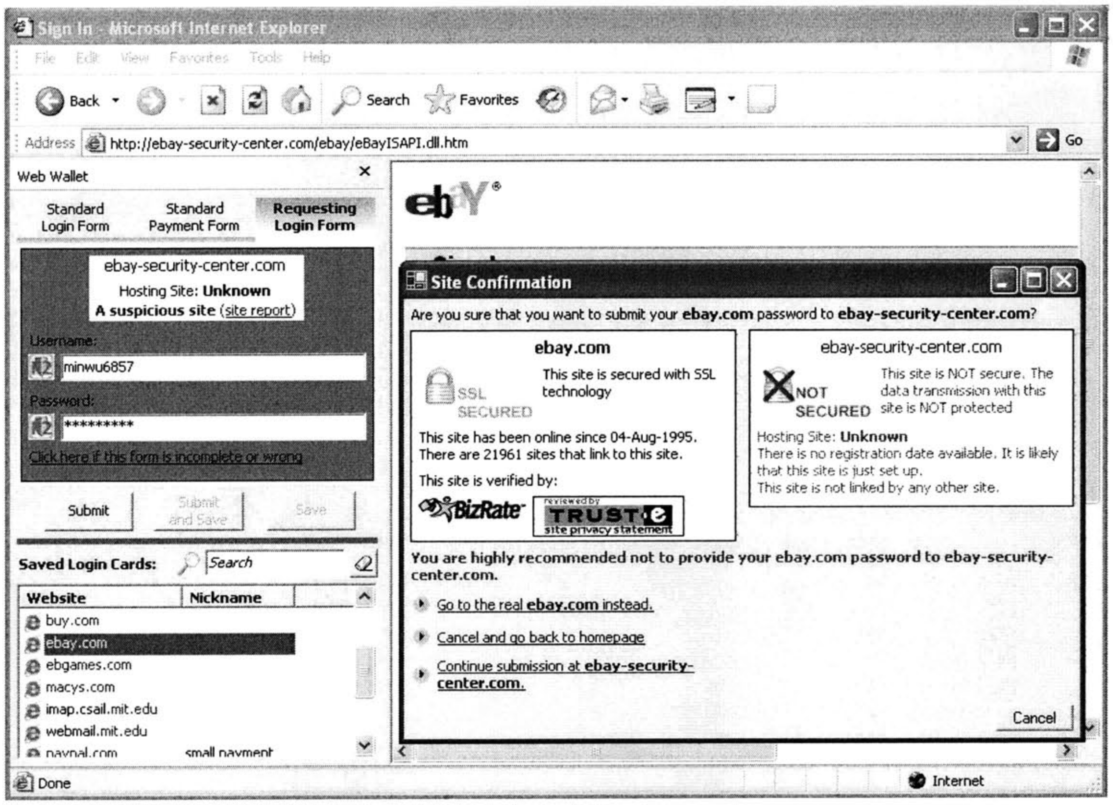
The user has to go through the list to choose his intended site. If the user's choice is different from the current site, Web Wallet warns about the discrepancy by showing the site reputation reports of the two sites side by side, and is given an alternative safe link to his intended site (figure 5-15(b)). The same warning will show when the user chooses a stored login card from the card folder, but the current site does not match the stored card (figure 5-14).

Even when the user's intended site is different from the current site, the dialog still includes an option for the user to continue at the current site. A user should be the final decision maker about his own online interaction. Web Wallet can effectively warn him if it decides that he is in danger but it can not make the decision for him, because the Web Wallet's decision cannot be proved to be always 100% correct.

If the user's choice is the same as the current site, the user will also see the site report explaining why this site is not verified before he continues with the submission (figure 5-15(c)).

5.2.5 Negative visual cue

Web Wallet includes a negative visual cue to indicate that submitting data at web forms is potentially dangerous. When a user types a character into a web page, the typed character is quickly zoomed out from the typing location to the center of the screen, semi-transparently (figure 5-16). By zooming the characters, this cue is expected to get the user's attention. Moreover, users are expected to feel uncomfortable when they are typing a password or a credit card number at a web page and the sensitive characters is flying out. Thus using Web Wallet to submit sensitive information will hopefully feel a more natural.



http://ebay-security-center.com/ebay/eBayISAPI.dll.htm

ebay-security-center.com
Hosting Site: **Unknown**
A suspicious site (site report)

Username: minwu6857
Password: *********

Click here if this form is incomplete or wrong

Website	Nickname
buy.com	
ebay.com	
ebgames.com	
macys.com	
imap.csail.mit.edu	
webmail.mit.edu	
paypal.com	small payment

Are you sure that you want to submit your **ebay.com** password to **ebay-security-center.com**?

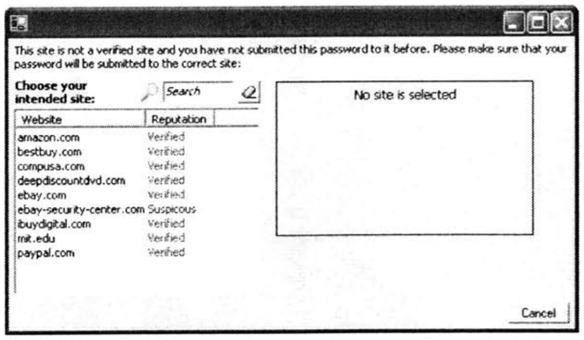
ebay.com
SSL SECURED
This site is secured with SSL technology
This site has been online since 04-Aug-1995.
There are 21961 sites that link to this site.
This site is verified by:
BizRate
reviewed by TRUSTe site privacy statement

ebay-security-center.com
NOT SECURED
This site is NOT secure. The data transmission with this site is NOT protected
Hosting Site: **Unknown**
There is no registration date available. It is likely that this site is just set up.
This site is not linked by any other site.

You are highly recommended not to provide your ebay.com password to ebay-security-center.com.

- Go to the real **ebay.com** instead.
- Cancel and go back to homepage
- Continue submission at **ebay-security-center.com.**

Figure 5-14: User chooses the stored eBay card at a simulated eBay-phishing site


This site is not a verified site and you have not submitted this password to it before. Please make sure that your password will be submitted to the correct site:

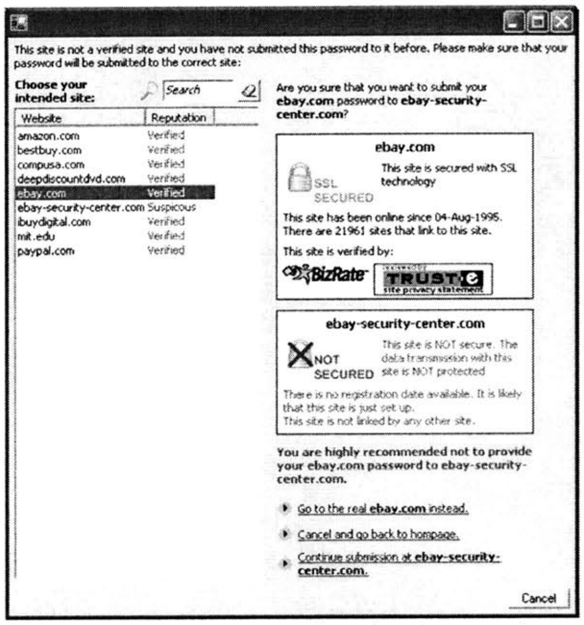
Choose your intended site: Search

Website	Reputation
amazon.com	Verified
bestbuy.com	Verified
compusa.com	Verified
deepdiscountdvd.com	Verified
ebay.com	Verified
ebay-security-center.com	Suspicous
ibuydigital.com	Verified
mit.edu	Verified
paypal.com	Verified

No site is selected

Cancel

(a) site confirmation dialog waiting for the user's intended site


This site is not a verified site and you have not submitted this password to it before. Please make sure that your password will be submitted to the correct site:

Choose your intended site: Search

Website	Reputation
amazon.com	Verified
bestbuy.com	Verified
compusa.com	Verified
deepdiscountdvd.com	Verified
ebay.com	Verified
ebay-security-center.com	Suspicous
ibuydigital.com	Verified
mit.edu	Verified
paypal.com	Verified

Are you sure that you want to submit your **ebay.com** password to **ebay-security-center.com**?

ebay.com

SSL SECURED This site is secured with SSL technology

This site has been online since 04-Aug-1995.
There are 21961 sites that link to this site.

This site is verified by:

ebay-security-center.com

NOT SECURED This site is NOT secure. The data transmission with this site is NOT protected

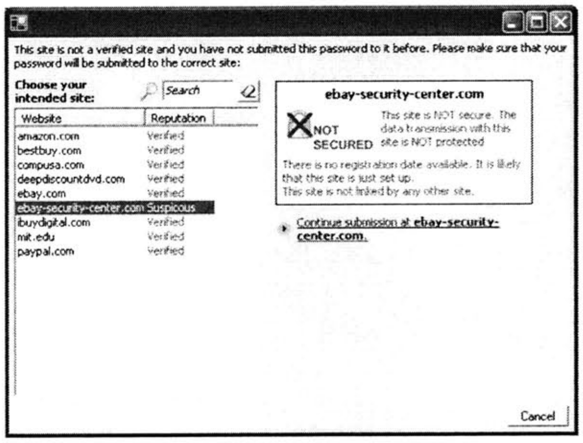
There is no registration date available. It is likely that this site is just set up.
This site is not linked by any other site.

You are highly recommended not to provide your ebay.com password to ebay-security-center.com.

- Go to the real **ebay.com** instead.
- Cancel and go back to hompage.
- Continue submission at **ebay-security-center.com.**

Cancel

(b) the user's intended site is different from the current site


This site is not a verified site and you have not submitted this password to it before. Please make sure that your password will be submitted to the correct site:

Choose your intended site: Search

Website	Reputation
amazon.com	Verified
bestbuy.com	Verified
compusa.com	Verified
deepdiscountdvd.com	Verified
ebay.com	Verified
ebay-security-center.com	Suspicous
ibuydigital.com	Verified
mit.edu	Verified
paypal.com	Verified

ebay-security-center.com

NOT SECURED This site is NOT secure. The data transmission with this site is NOT protected

There is no registration date available. It is likely that this site is just set up.
This site is not linked by any other site.

- Continue submission at **ebay-security-center.com.**

Cancel

(c) the user's intended site is the current site

Figure 5-15: Web Wallet's site confirmation dialog

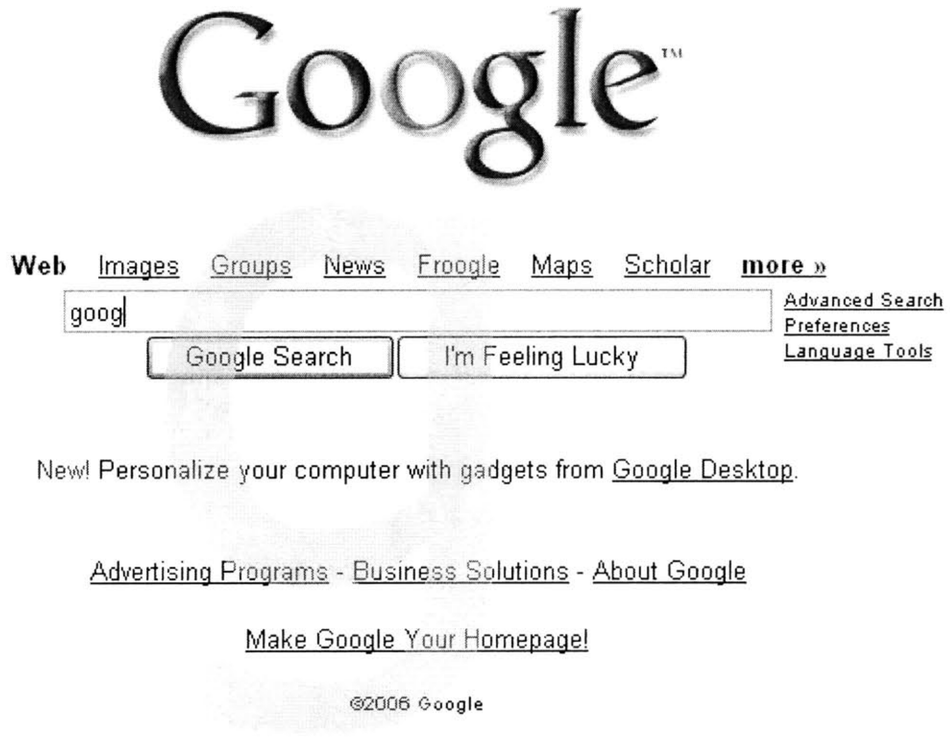

Google
Web Images Groups News Froogle Maps Scholar more »
goog
Google Search
I'm Feeling Lucky
Advanced Search
Preferences
Language Tools
New! Personalize your computer with gadgets from Google Desktop.
Advertising Programs - Business Solutions - About Google
Make Google Your Homepage!
©2006 Google

Figure 5-16: Web Wallet's negative visual cue when a user is typing at the web page

Chapter 6

Sensitive-Input Detection Algorithm

In order to prevent users from typing sensitive information into web pages, Web Wallet searches every page for web forms that ask for sensitive information, such as login information or credit card information, and disables their sensitive inputs. This chapter introduces the sensitive-input detection algorithm.

6.1 Definition of sensitive data

Current Web Wallet can protect two types of sensitive information: login and credit card.

Login information includes two types of sensitive data, the username and the password. Because users can register new accounts and update their account information online, each sensitive type has three states: *existing*, *new* and *confirm*. Figure 6-1 demonstrates different login-related data types and their states. The *existing* state is the default state and is omitted from the annotation. Because Web Wallet can save the user's login submission as a login card, it is very important for it to correctly detect not only the sensitive type but also the state. For example, Web Wallet should not save a confirmed password as a second password for an account when it deals with the registration form in figure 6-1(b), and it should save the new password but

Index	**Sensitive type (state)**	**Notes**
1	Login.username	
2	Login.username (new)	
3	Login.username (confirm)	
4	Login.password	
5	Login.password (new)	
6	Login.password (confirm)	
7	Card.type	
8	Card.number	
9	Card.expiration.ymd.year	
10	Card.expiration.ymd.month	
11	Card.verification	The verification code of a credit card
12	Card.name	Cardholder's name
13	Card.pin	Most of the time, only a debit card asks for a PIN

Table 6.1: Sensitive types (and state) for login information and credit card information

not the old existing password when it deals with the update form in figure 6-1(c).

Credit card information includes seven types of sensitive data: card type, card number, card expiration month, card expiration year, card verification code, cardholder's name and card PIN. Some types, like card type and cardholder's name, are not sensitive by themselves. But since they are always requested together with the credit card number, it is better to detect them so that the requested credit card information can be filled all at once from Web Wallet. Figure 6-2 demonstrates different credit-card-related data types. Since users never create their new credit card online by specifying a new credit card number or verification code for that card, all the credit card types have only one state, *existing*. Table 6.1 lists the Web-Wallet-protected sensitive data types and their states.

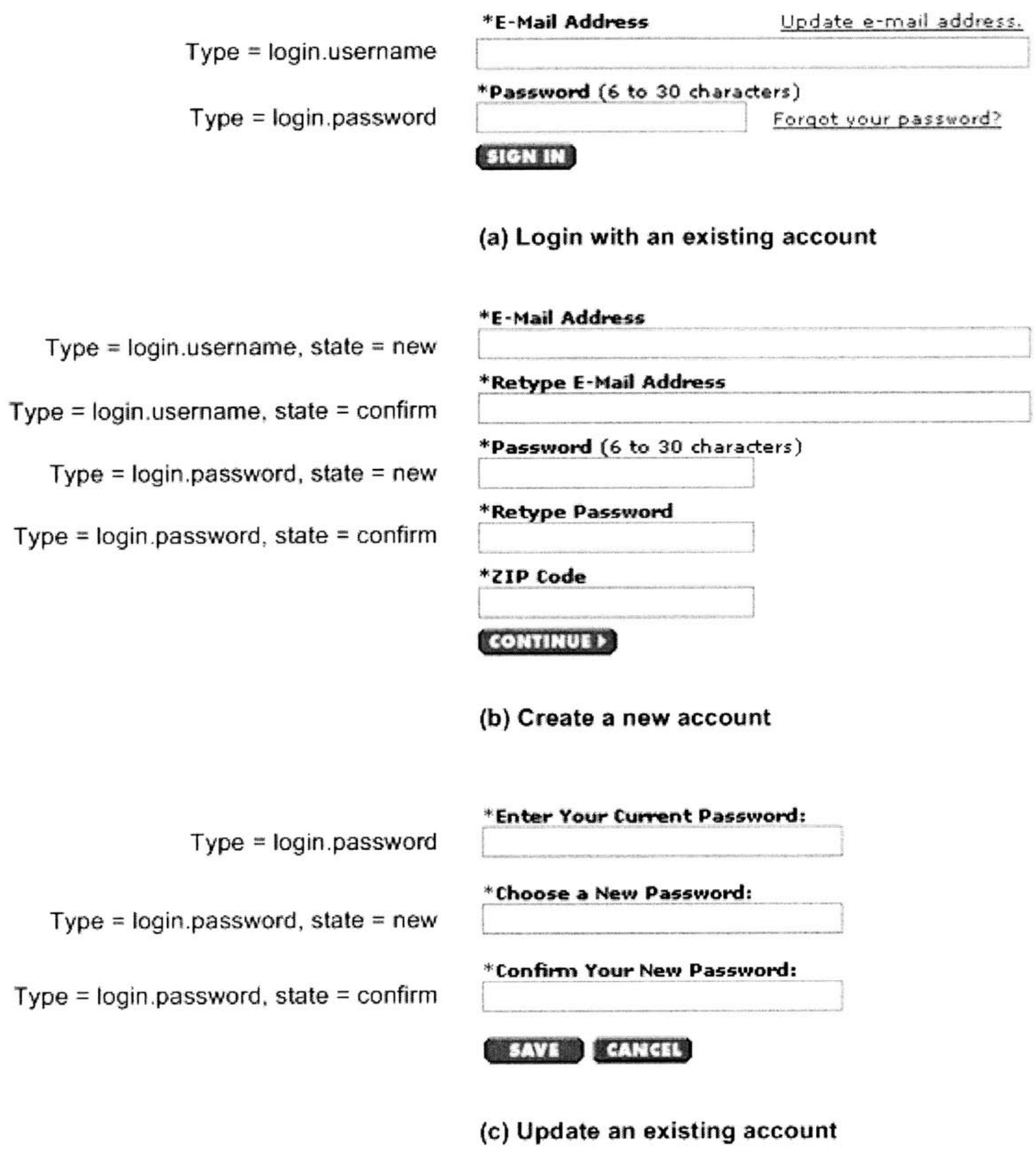

Type = login.username
Type = login.password
*E-Mail Address
Update e-mail address.
*Password (6 to 30 characters)
Forgot your password?
SIGN IN
(a) Login with an existing account
Type = login.username, state = new
Type = login.username, state = confirm
Type = login.password, state = new
Type = login.password, state = confirm
*E-Mail Address
*Retype E-Mail Address
*Password (6 to 30 characters)
*Retype Password
*ZIP Code
CONTINUE
(b) Create a new account
Type = login.password
Type = login.password, state = new
Type = login.password, state = confirm
*Enter Your Current Password:
*Choose a New Password:
*Confirm Your New Password:
SAVE
CANCEL
(c) Update an existing account

Figure 6-1: Sensitive types and states for login information

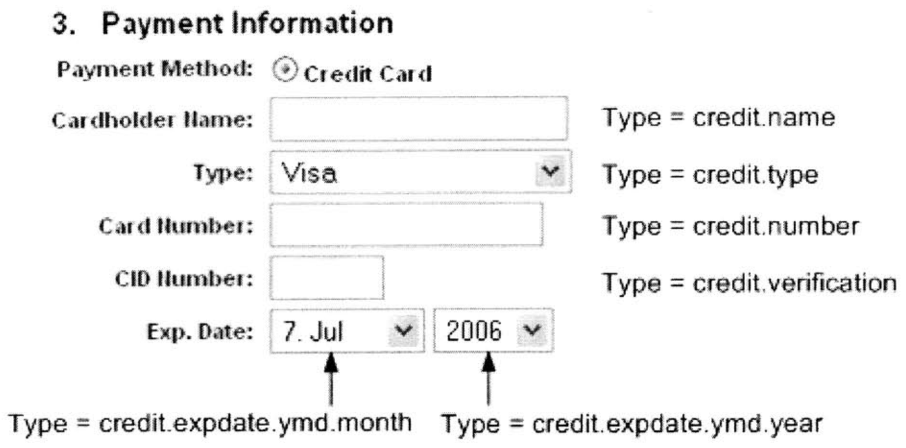


Figure 6-2: Sensitive types for credit card information

6.2 Sensitive-input detection algorithm

The outline of the algorithm is as follows: first, all the inputs in the page are grouped by the HTML form in which they appear. Three types of HTML elements are targeted by the algorithm: INPUT elements with "text" type, INPUT elements with "password" type, and SELECT elements. Henceforth all three types of elements will be referred to simply as *inputs*.

For each input in a group, up to four pieces of information are collected:

- *Input label*: A visible text string associated with the input that tells users what type of data the input is requesting
- *Input name*: The "name" property of the input that is bound with the input value when the form is submitted
- *Input type*: The HTML element type, either a "text" INPUT, a "password" INPUT, or a SELECT
- *Input options*: If the input is a SELECT element, its OPTION elements.

Based on these four pieces of information, and the ordinal order that each input appears in the HTML form, a Hidden Markov Model (HMM) is applied to the sorted

<A>	<BIG>	<FONT>	<SAMP>	<SUB>
<ABBR>	<CITE>	<I>	<SMALL>	<SUP>
<ACRONYM>	<CODE>	<KBD>	<SPAN>	<TT>
<B>	<DFN>	<OPTION>	<STRIKE>	<U>
<BASEFONT>	<EM>	<S>	<STRONG>	<VAR>

Table 6.2: HTML tags for non-partitioning elements

list of inputs in order to determine the relations among them and the structure of the form request by assigning the most possible sensitive type (and state) to each input. The HMM is trained using a set of web forms whose input elements that are manually assigned with sensitive types and states. Finally a set of semantic rules are applied to each group to refine the derived structure.

6.2.1 Algorithm to find the input label

The input label is the visible text associated with the input in a form. It can be found heuristically as follows.

The targeted input element is treated as an HTML DOM node. The previous sibling of this node will be the potential label node. In the potential label node, its text blob is searched. A *text blob* is a visible string of content delimited by the opening and closing tags of a partitioning HTML element, which is an ancestor of a group of text nodes that appear together in the rendered HTML page. [5]. The HTML tags that do not represent partitioning elements in this algorithm are listed in table 6.2. If the text in the text blob contains letters or digits, the text is the input label and the search stops. Otherwise the next potential label node is the previous sibling of the current label node. If there are no previous siblings, the previous sibling of its parent is used instead. This search will end when the potential label node is the form itself.

6.2.2 Tokenization

As mentioned previously, each input has up to four pieces of information that can be used to determine its sensitive type. These pieces of information are tokenized first. The input label is tokenized by the characters that are not letters and digits. The input name is tokenized by non-letter characters and based on the uppercase-lowercase switch. Each option of a SELECT element is tokenized by the characters that are not letters and digits. The label tokens, the name tokens and the option tokens are then converted to lowercase and prefixed with "L:", "N:" and "O:" correspondingly. Finally, the input type is another token: the "text" INPUT element and the SELECT element has a "P:nonpassword" token and the "password" INPUT element has a "P:password" token. Figure 6-3 shows a tokenization example. It also shows an example on how to find a label input, as discussed in section 6.2.1.

6.2.3 Applying Hidden Markov Model

Each input can be assigned to 14 classes, including the 13 sensitive types in table 6.1 and a class representing all other (insensitive) inputs. As shown in figure 6-4, each input is a random variable in the Markov chain. The derived tokens are the observable output for each random variable. In order to find the most likely class sequence, two types of probabilities have to be computed from the training data: the local probability of each input to each class $\Pr(T|s_k)$, and the transition probabilities between consecutive inputs $\Pr(s_{k+1}|s_k)$.

Local probabilities of each input

One-vs-all-but-one Multinomial Naive Bayes approach [79] is used to decide the probability of each input to each class.

Multinomial Naive Bayes (MNB) models the distribution of tokens of an input as a multinomial [48]. An input is treated as a sequence of tokens and it is assumed that each token position is generated independently of each other. The probability of an input to be in a class is a product of the probability of each of its token occurs

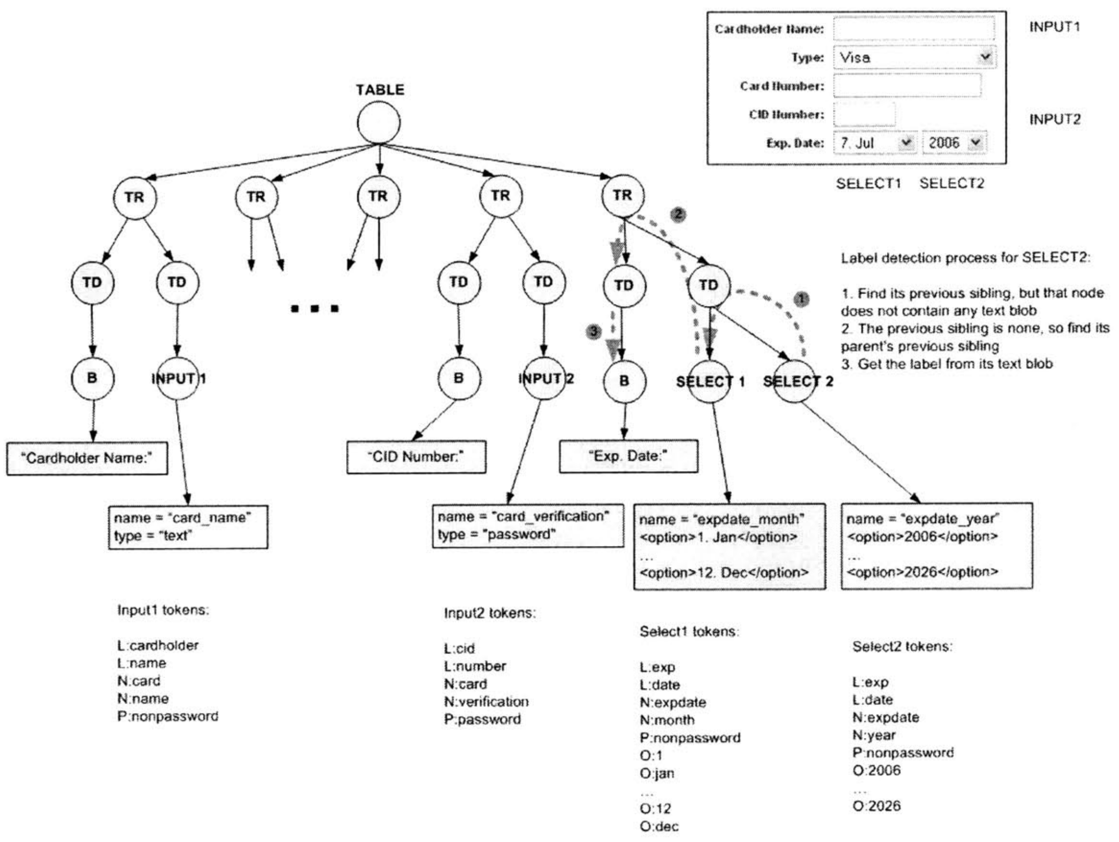


Figure 6-3: An example about how to generate tokens for an input

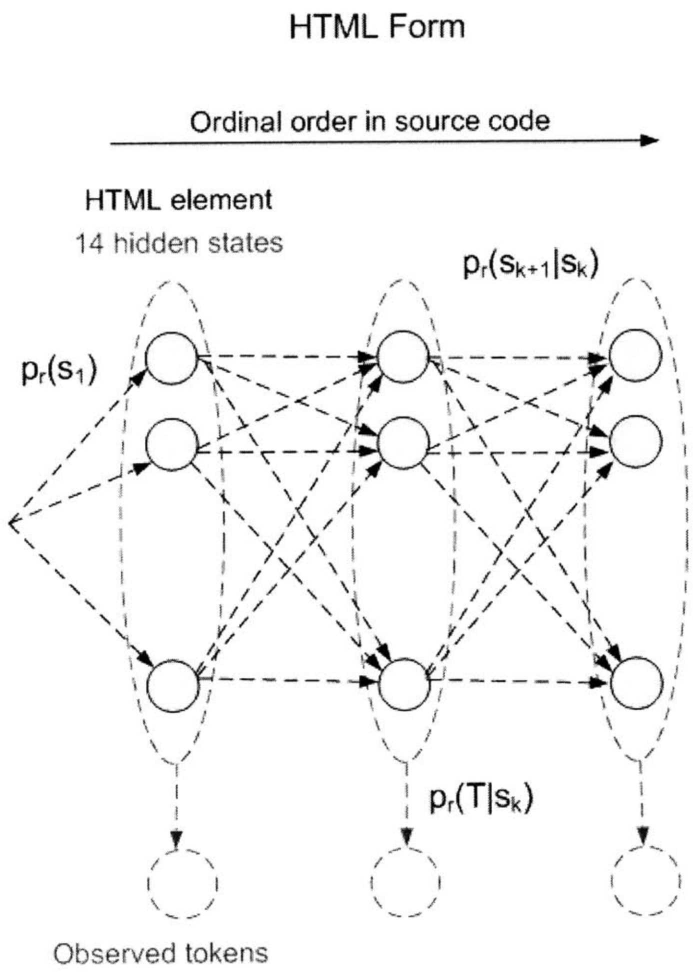


Figure 6-4: Apply a Hidden Markov Model to a input list in a form

in that class:

$$\Pr(t|s) = \frac{N_{st} + 1}{N_s + |V|}$$

Here N_{st} is the number of times that token t appears in class s in the training data, N_s is the total number of tokens in class s in the training data, and V is the vocabulary of tokens generated from the training data.

Complement Naive Bayes (CNB) is proposed to reduce one type of errors of MNB [55]. That is, more training examples for one class than another can cause the classifier to prefer the more highly trained class. CNB computes the probability of an input to be in a class using the training data from all classes except that class:

$$\Pr(t|\neg s) = \frac{N_{\neg st} + 1}{N_{\neg s} + |V|}$$

Here $N_{\neg st}$ is the number of times that token t appears in classes other than class s in the training data, $N_{\neg s}$ is the total number of tokens in classes other than class s in the training data.

One-vs-all-but-one Multinomial Naive Bayes computes the difference between the MNB result and the CNB result:

$$\Pr(t|\hat{s}) = \Pr(t|s) - \Pr(t|\neg s)$$

The log probability for an input to a class is the sum of each token's count times its one-vs-all-but-one Multinomial Naive Bayes probability:

$$\log \Pr(T|\hat{s}) = \sum_{t \in T} d_t (\log \Pr(t|s) - \log \Pr(t|\neg s))$$

Here T is the set of the tokens for this input, d_t is the count of token t in T.

Local probability refinement

The local probabilities of each input to different classes are refined based on the HTML element type. If an element is a SELECT element, then it can not be an input for password, credit card number, credit card verification code, credit card PIN and cardholder's name. Therefore, the log probabilities of the following seven classes (login.password, login.password.new, login.password.confirm, credit.number, credit.verification, credit.pin, credit.name) are set equal to $-\infty$ if the input is a SELECT element. Moreover, if an element is a "text" INPUT element, then it can not be an input for the password, and the log probabilities of the three password classes (login.password, login.password.new, login.password.confirm) are set equal to $-\infty$.

As mentioned before, both login.username and login.password has three states: existing, new, and confirm. In the training data, for each type, these three states share many common keywords like "login", "username" or "password" but only differentiate in a few unique keywords like "new" to indicate that new state and "confirm" or "verify" to indicate the confirm state. The effects of the unique keywords are lessened by the majority of the common keywords. On the other hand, the unique keywords are very important to differentiate the states. Therefore, keyword matching is used to modify the probabilities for different states. Consider login.username as an example: for each input, $\Pr_{\max}(T|\hat{s}_{\text{username}})$ and $\Pr_{\min}(T|\hat{s}_{\text{username}})$ are the maximal and the minimal probabilities among the three username-related classes. That is,

$$\begin{aligned}\Pr_{\max}(T|\hat{s}_{\text{username}}) = \max(&\Pr(T|\hat{s}_{\text{username, existing}}), \Pr(T|\hat{s}_{\text{username, new}}),\\ &\Pr(T|\hat{s}_{\text{username, confirm}}))\\ \Pr_{\min}(T|\hat{s}_{\text{username}}) = \min(&\Pr(T|\hat{s}_{\text{username, existing}}), \Pr(T|\hat{s}_{\text{username, new}}),\\ &\Pr(T|\hat{s}_{\text{username, confirm}}))\end{aligned}$$

If the input tokens contain the keywords for the new (confirm) state, the probability of this input to the username new (confirm) class is $\Pr_{\max}(T|\hat{s}_{\text{username}})$ and the probabilities to the other two classes are $\Pr_{\min}(T|\hat{s}_{\text{username}})$. The login.password changes its probabilities of different states in the same way. All the state related keywords are listed in table 6.3.

Transition probabilities of Hidden Markov Model

Given the probabilities of each input to all the 14 classes are available, the Hidden Markov Model is applied to the input list. In figure 6-4, $\Pr(s_k = i) = N_i/N$ and

State	**Keywords**
New	"create", "new"
Confirm	"reenter", "repeat", "verify", "confirm", "again", "retype", "check", "second"

Table 6.3: Keywords for the new state and the confirm state

$\Pr(s_{k+1} = j | s_k = i) = N_{ij}/N_{\text{tr}}$ are estimated from the training data, where N_i is the total number of the input in class i and N is the total number of the inputs in the training data; N_{ij} is the total number of the consecutive inputs with first input is in class i and the second input is in class j ($N_{ij} = 1$ if no such consecutive inputs exist in the training data) and N_{tr} is the total number of the consecutive inputs. The Viterbi algorithm [26] finds the most likely path in HMM and the most likely class is assigned to each input accordingly.

6.3 Evaluation of Hidden Markov Model

465 forms from 78 sites were collected, which contains four parts: (1) 20 tested sites in the Web Wallet prototype evaluation, (2) 26 sites that are recently attacked by phishing (in June and July 2006) based on the Anti-Phishing Working Group's phishing archive, (3) about 10 sites that I personally used during the collection period (about a week), and (4) about 20 sites that were selected from the Google search result for "login" and "credit card payment". I manually assigned sensitive types and states to individual inputs in the forms. Among the 465 forms, 170 of them (sensitive forms) contain sensitive inputs and the other 295 (insensitive forms) do not. In order to do cross validation, half of the 170 sensitive forms and half of the 295 insensitive forms were randomly selected to be the training set and the remaining half of the forms were the test set. The cross validation was run 10 times. Each time a different random partition was chosen. The results are averaged over the 10 runs.

Recall and Precision are defined as:

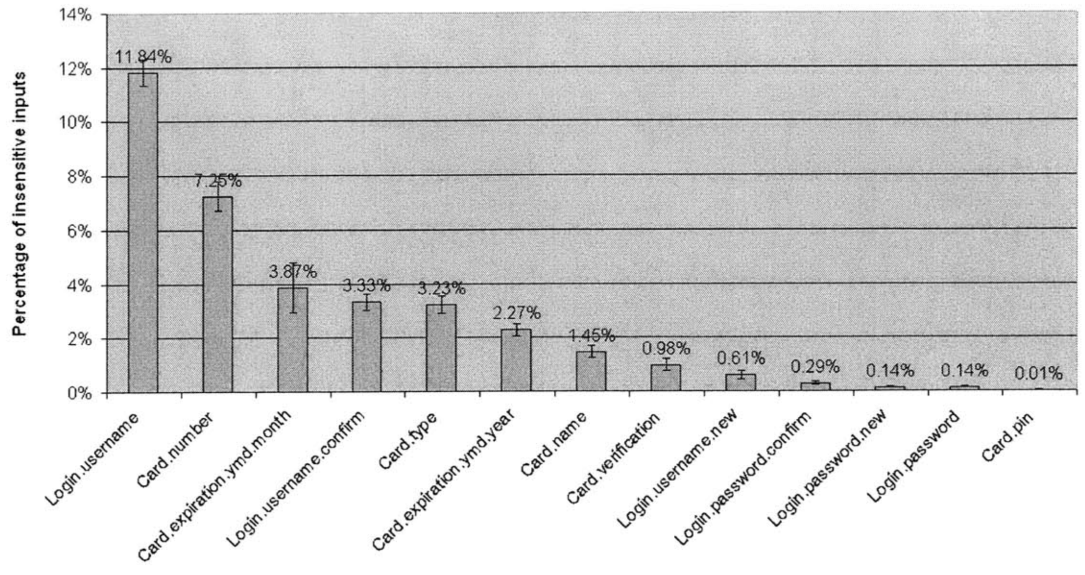


Figure 6-5: Percentage of insensitive inputs mistakenly detected as sensitive

$$\text{Recall} = \frac{\#\text{ of correct detected sensitive inputs in terms of both the type and the state}}{\#\text{ of sensitive inputs}}$$

$$\text{Precision} = \frac{\#\text{ of correct detected sensitive inputs in terms of both the type and the state}}{\#\text{ of detected sensitive input}}$$

Learning using HMM has a very good recall (94.5%) but a poor precision (44.4%). Poor precision means that many insensitive inputs are mistakenly detected as sensitive. As shown in figure 6-5, login username is the sensitive type that insensitive inputs are most wrongly assigned. One reason is that many web sites use the email address as the login username and thus "address" is a keyword for username type Inputs asking for postal addresses are thus sometimes treated as a username.

Both poor recall and poor precision prevent Web Wallet from being an effective anti-phishing solution. With poor recall, many sensitive inputs in web pages fail to be disabled, and thus users cannot be successfully trained to depend on Web Wallet to submit sensitive information. Poor precision makes Web Wallet incorrectly disable insensitive inputs, so users have to open Web Wallet in order to fill in these inputs. If the site is not verified, users are even unnecessarily asked to confirm their intended site. This definitely hurts the usability of the web pages. Moreover, both poor recall and poor precision make users not fully trust the Web Wallet protection.

6.4 Semantic rules to improve accuracy

In order to improve the accuracy of the sensitive-input detection algorithm, eight semantic rules are applied to the most likely state sequence of the input list computed using HMM.

Web Wallet not only needs to correctly detect sensitive inputs in a given form, but also needs to group related sensitive inputs together so that Web Wallet can fill them at once. A set of related sensitive inputs that can be filled together is called a protected area. Because the current Web Wallet can only protect login information and credit card information, a protected area is either a login area that can only contain login types as username and password, or a credit-card area that can only contain the seven credit card sensitive types. Hence semantic rule 1 says: *Each protected area can only contain inputs with sensitive types either from the login category or the credit card category but not both.*

For each protected area, different types have different security priorities. For example, since the password is the secret for login and the username is just an identity but not a secret, any login area that does not contain a password input can be discarded. The same is true for the credit card number and PIN inputs for the credit-card area. Semantic rule 2 says: *If a login area does not contain a password or if a credit-card area contains neither a card number input nor a card PIN input, discard this area.*

It is a common case that for authentication the username is asked before the password. No web form has been seen that asks for the password and then the corresponding username. But there are cases that a form contains two login areas, one for login and the other for registration. Hence semantic rule 3 says: *When a username input is going to be added to a login area that already contains a password, create a new login area and add that username input into the new area.*

Because each credit card has only one card type, expiration date, verification code, cardholder's name and PIN, semantic rule 4 says: *When a credit-card-related type is going to be added to a credit-card area and that type is not card number and that type is already in the area, create a new credit area and add that type into the new area.* A card number input is a special case because some sites use several inputs (usually four) for a single card number. So semantic rule 4.1 is added: *If it is a card number input and that type is already in the area and the last input is not a card number, create a new credit area and add that type into the new area.*

Credit card expiration date and verification code are card-number dependent. That is, without first specifying a card number, it is meaningless to ask for the card expiration date and the verification code. Semantic rule 5 says: *When an input for expiration date or verification code is going to be added to a credit-card area and that area does not have a card number yet, treat the input as insensitive and discard it.*

An HTML INPUT element with the "password" type is a strong indicator that the input's sensitive type is password, but it is not always true. A credit card PIN and a credit card verification code are sometimes also protected by the "password" type so that their input value can be visually masked from shoulder surfing. Because the data set contains many more password inputs than protected card verification inputs and card PIN inputs, Naive Bayes has a strong tendency to treat every password field as the password type. The semantic rule 6 says: *When a password input is going to be added to a credit-card area, find out if this input is actually an input for either the credit card PIN or the credit card verification code by using the PIN-keyword checking and the verification-keyword checking. If yes, change its type accordingly. Then use rule 5 if the type is verification code.* (Keywords for different sensitive types are listed in table 6.4.)

For each form that asks for credit card information, the card number is always requested. Other pieces of data, like the cardholder's name or the verification code are not requested so often. As a result, the data set contains more credit card number inputs than other types. This makes the Naive Bayes prefer the card number type. Here is the semantic rule 7: *When an input of the card number type is going to be*

Sensitive data type	Keywords
Card.verification	“cid”, “verification”, “cuuv”, “id”, “cvv”, “security”, “identification”
Card.pin	“pin”
Card.name	“name”, “holder”, “owner”
Credit-card area	“credit”, “debit”, “atm”, “account”, “card”

Table 6.4: Keywords for the sensitive data type

added to a credit-card area, first find out if this input is actually an input for the cardholder's name or the credit card PIN or the credit card verification code by using the PIN-keyword checking and the verification-keyword checking. If yes, change its type accordingly. Then apply rule 5 if the type is verification code.

“Number” is a keyword of the type of the credit card number. Naive Bayes sometimes treats inputs asking for a phone number or a fax number as a card number. As shown in figure 6-5, card number is the sensitive type that insensitive inputs are second-most wrongly assigned. Moreover, since the card number is the critical information for the credit card area, rule 2 can not correct this mistake. Hence semantic rule 8 says: *In a credit-card area, if all its input tokens fail with the credit-keyword checking, discard this area.*

6.5 Final evaluation

The semantic rules not only dramatically increase the precision (one-tail t(18) = -19.49, p < 1e-13) but also improve the recall (one-tail t(16) = -6.53, p < 1e-5), as shown in figure 6-6.

Figure 6-7 shows the recalls and precisions of the algorithm to detect the login-related inputs and the credit-card-related inputs. Both the recall and the precision for the login-related inputs are very high. But the precision for the credit-card-related inputs is not high (about 80%). The main reason is that the detection algorithm

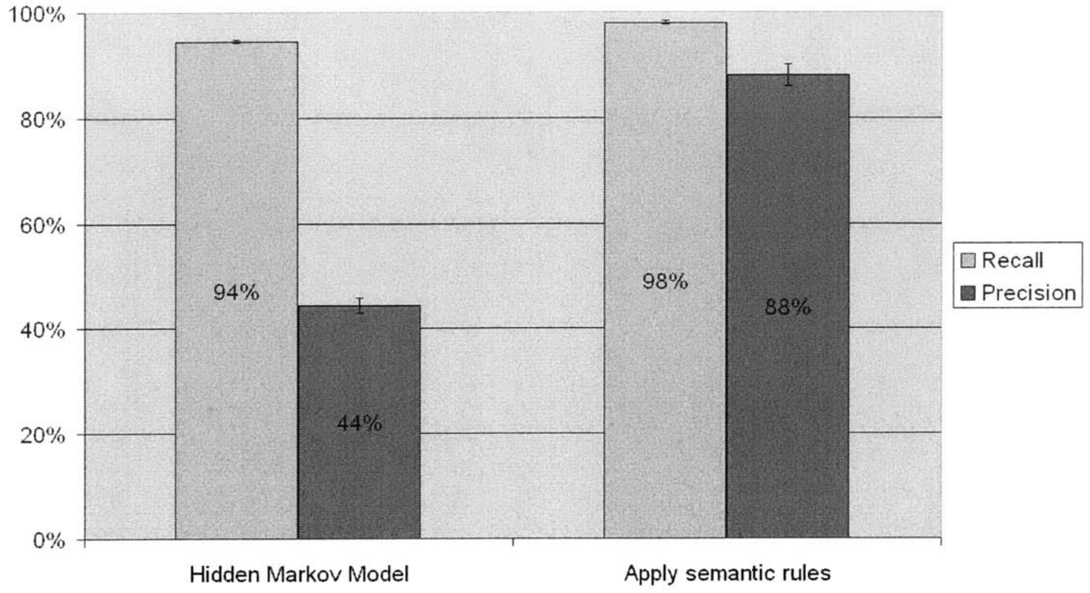


Figure 6-6: Recall and precision of the sensitive-input detection algorithm

treats the inputs that ask for gift card information as the credit card inputs (figure 6-8). I did not specify them as the credit card inputs during the data collection period. On the other hand, it is not a serious detection error. Some users may want to store their gift cards into their Web Wallet's card folder. By default, Web Wallet does not designed to include the protection of the gift cards because they usually have an expense limit and are unlikely to cause significant financial loss as the credit card.

Figure 6-9 shows recall and precision with different cross-validation ratios of test to training data. Both the recall and the precision do not change much. The algorithm is pretty stable and does not depend on the size of the training data much. It also reflects the fact that the web forms for login information and credit card payment information from different web sites are generally consistent.

6.6 Comparison with other form filling algorithms

There are many existing auto-fill software available. For example, Google toolbar (toolbar.google.com) can automatically fill some web forms that request certain types

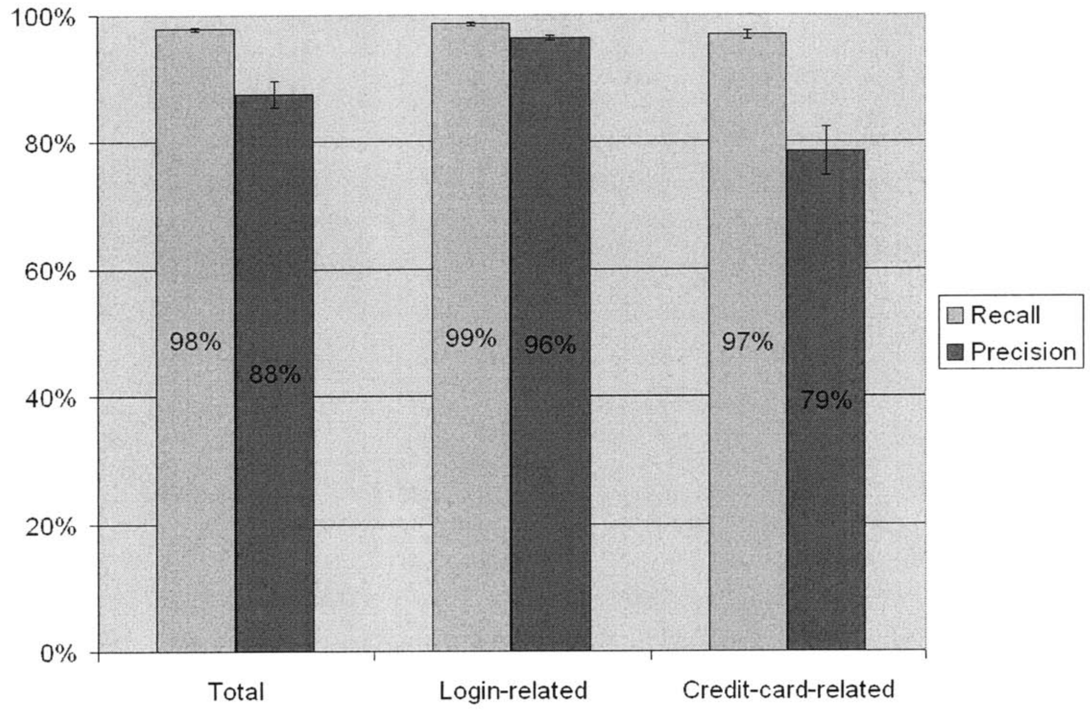
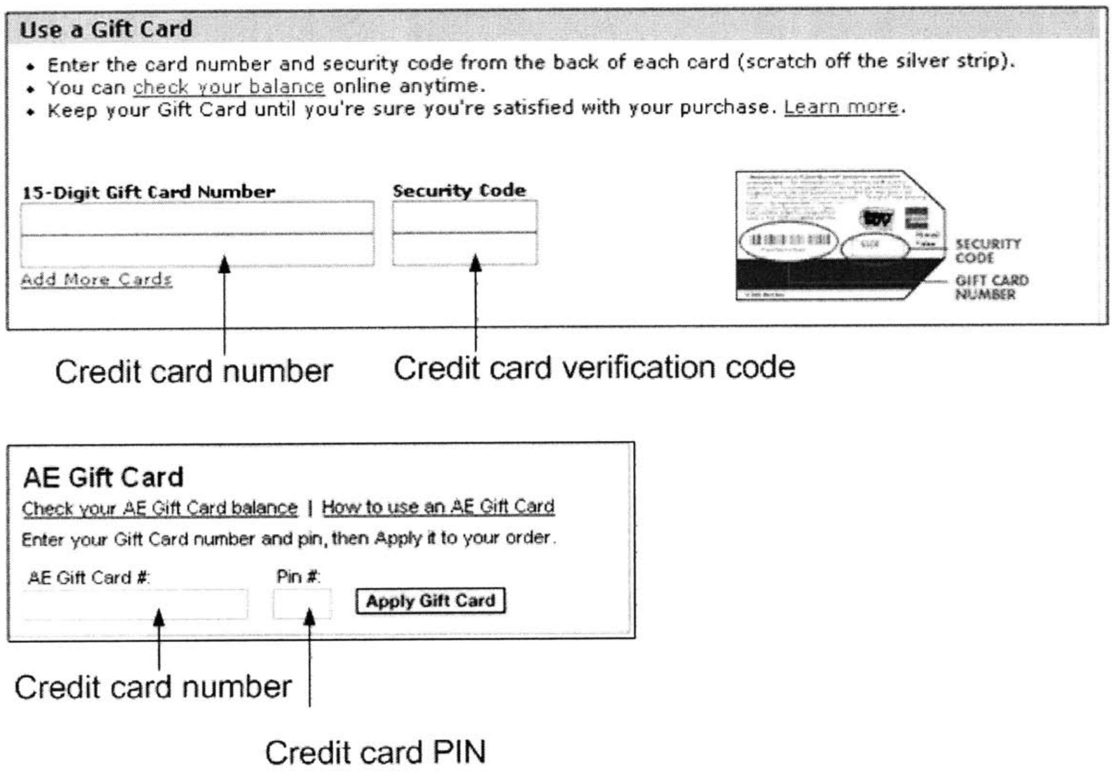


Figure 6-7: Recall and precision of the algorithm to detect the login-related and the credit-card-related inputs



Use a Gift Card

- Enter the card number and security code from the back of each card (scratch off the silver strip).
- You can check your balance online anytime.
- Keep your Gift Card until you're sure you're satisfied with your purchase. Learn more.

15-Digit Gift Card Number | Security Code

Add More Cards

AE Gift Card

Check your AE Gift Card balance | How to use an AE Gift Card

Enter your Gift Card number and pin, then Apply it to your order.

AE Gift Card #: | Pin #:

Figure 6-8: Detection algorithm treats the gift card request as the credit card request

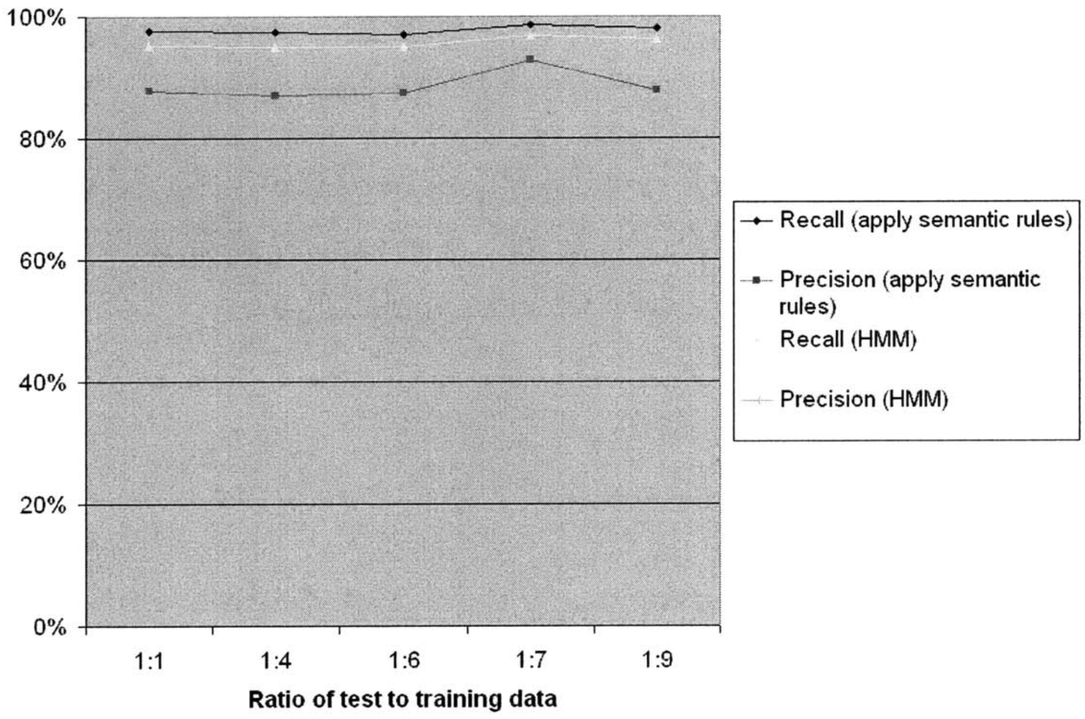


Figure 6-9: Recalls and precisions with different cross validation ratios of test to training data

of personal information, including a user's name, email address, phone number, physical address and credit card information. Google toolbar auto-fill feature mainly depends on the field name of the HTML input elements to decide if this element can be filled. RoboForm (www.roboform.com) is another password manager and web form filler. It claims to be the most precise form filler on the market. It uses "Artificial Intelligence techniques" to detect the HTML inputs that can be filled. [56] (I have personally contacted the RoboForm development team in order to find out its form filling algorithm in detail. But unfortunately, I was told that it is a trade secret.)

Web Wallet's sensitive-input detection algorithm is tested against these two auto-fill applications. Only the credit-card related inputs were tested, because (1) Google toolbar does not store site passwords at all, and (2) RoboForm does not support all login-related forms as Web Wallet does. For example, RoboForm cannot remember new login account information that is typed into a registration form.

6.6.1 Results

Thirty five credit-card related forms were collected in the following two ways: (1) 15 forms that were selected from the Google search result for "credit card submission" and (2) 20 credit card payment form that were from 20 shopping sites selected in the shopping category at dmoz.org.

Among the 35 forms, there are 144 credit-card related inputs valid to Google toolbar and 160 credit-card related inputs valid to Web Wallet and RoboForm. The difference is because Google toolbar does not store the credit card verification code and thus cannot fill the corresponding input.

Figure 6-10 shows both the recalls and the precisions of the three detection algorithm of Web Wallet, Google toolbar and RoboForm. The recalls are significantly different (single-factor ANOVA test: $p < 1\text{-e}7$) but the precisions are not (single-factor ANOVA test: $p = 0.07$). The recall of Web Wallet is significantly higher than the one of Google toolbar and is comparable with the one of RoboForm. The precision of Web Wallet is the lowest among the three, but the differences are not large. RoboForm *does* have very high accuracy (highest recall and precision). Web

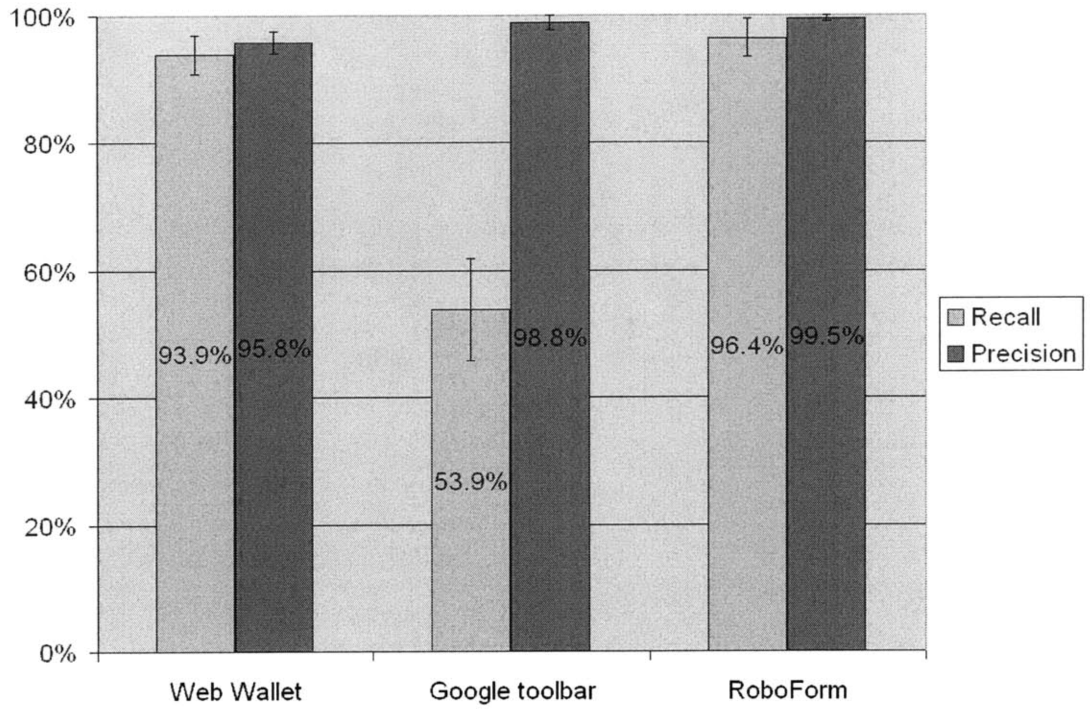
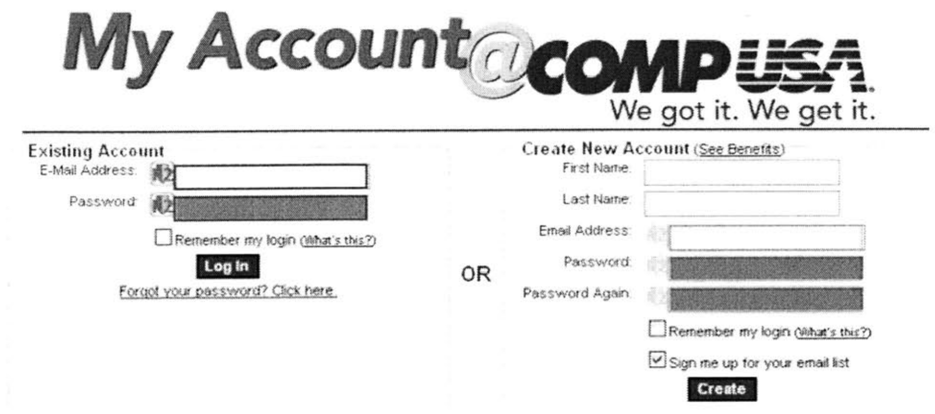


Figure 6-10: Recalls and precisions of the sensitive-input detection algorithms of Web Wallet, Google toolbar and RoboForm

Wallet's algorithm is in general more accurate than the Google toolbar's algorithm because Web Wallet depends not only on the field name of individual input element but also on the inter-input relationships by applying Hidden Markov Model and a set of semantic rules to a whole form. Improving Web Wallet detection accuracy or integrating RoboForm form filling algorithm into Web Wallet is a question for future work.

6.7 Sensitive types specified by web site

To further improve detection accuracy, web sites can add some attributes to its input elements, indicating which inputs are sensitive and what type of sensitive data the input is asking. The most important attribute is the one to indicate the data type. If


My Account@COMPUSA
We got it. We get it.
Existing Account
E-Mail Address:
Password:
Remember my login (What's this?)
Log In
Forgot your password? Click here.
OR
Create New Account (See Benefits)
First Name:
Last Name:
Email Address:
Password:
Password Again:
Remember my login (What's this?)
Sign me up for your email list
Create

First (active) protected area:

```
<input
id="frm_email" name="frm_email" maxlength="80"
class="short" value="" sensitive="login.username"
sensitivearea="login">
<input
id="frm_password" name="frm_password"
maxlength="25" class="short"
type="password" sensitive="login.passwod"
sensitivearea="login">
```

Second protected area:

```
<input name="frm_email_address" id="frm_email_address" value=""
size="25" maxlength="80" class="short" type="text"
sensitive="login.username, new" sensitivearea="register">
<input
name="frm_passwordNew1" id="frm_passwordNew1" size="20"
maxlength="25"
class="short" type="password" sensitive="login.password, new"
sensitivearea="register">
<input name="frm_passwordNew2" id="frm_passwordNew2" size="20"
maxlength="25" class="short" type="password" sensitive="login.password,
confirm" sensitivearea="register">
```

Figure 6-11: Adding Web Wallet sensitive attributes to HTML elements

the data type is login.username or login.password, a state, either "new" or "confirm" may be added to the attribute. If there is more than one protected area in the web page, a second attribute may be added to indicate which area the sensitive input is included. Figure 6-11 shows how to add the Web Wallet attributes to an HTML element.

P3P [71] is planned to be integrated with Web Wallet so that all the defined private data types in P3P can be protected. One reason to choose P3P is that it already has a set of terminology for the private data. Another reason is that the P3P-enabled sites can easily add the Web Wallet attribute to their sensitive input fields.

Electronic Commerce Modeling Language (ECML) [17] specifies a standard set of HTML input names for payment-related information. For example, the input name for credit card number should be "Ecom_Payment_Card_Number". ECML version 2 is extended by the fields in a W3C P3P Note related to eCommerce [9]. The purpose of ECML is to make online transactions easily automated. Google toolbar supports ECML. Web Wallet supports ECML too. It first checks the HTML input element names with the standard ECML standard payment-related names before it goes through its own sensitive-input detection algorithm. Table 6.5 shows how the Web-Wallet protected login information and credit card information are defined by P3P and by ECML.

Protected sensitive type	**Corresponding P3P type (ECML standard name)**
Login.username	user.login.id (Ecom_User_ID)
Login.password	user.login.password (Ecom_User_Password)
Card.type	ecom.payment.card.type (Ecom_Payment_Card_Type)
Card.number	ecom.payment.card.number (Ecom_Payment_Card_Number)
Card.expiration.ymd.year	ecom.payment.card.expdate.ymd.month (Ecom_Payment_Card_ExpDate_Month)
Card.expiration.ymd.month	ecom.payment.card.expdate.ymd.year (Ecom_Payment_Card_ExpDate_Year)
Card.verification	ecom.payment.card.verification (Ecom_Payment_Card_Verification)
Card.name	ecom.payment.card.name (Ecom_Payment_Card_Name)
Card.pin	ecom.payment.card.pin (not specified)

Table 6.5: P3P data types (and ECML standard names) for login and credit card information

Chapter 7

Web Wallet Security Evaluation

Although Web Wallet is security software, it will be used by a human user. Therefore, before implementing a full-featured Web Wallet, I started with a prototype and ran a controlled user study to test both its usability and its effectiveness at preventing phishing attacks. The tested Web Wallet prototype only supports login information, and the backend is mostly hard-coded for the web sites used in the study. For example, the sensitive inputs asking for username and password are manually annotated and disabled.

The figures in this chapter show the user interface of the Web Wallet prototype, which is different from the interface of the current implementation, as shown in the figures in chapter 5.

7.1 Study design

The Web Wallet study used the same scenario as the anti-phishing toolbar study in chapter 3. Each subject was told to act as the personal assistant of John Smith. John Smith forwarded 20 emails to the subject and asked him to go to 20 different web sites, log in with his password, and add items to his wish list. Unlike the toolbar study, no tutorial about Web Wallet was included, since few users in the real world read tutorials. Web Wallet's interface must be self-explanatory.

Five of the 20 forwarded emails were attacks, with links leading the subject to phishing web sites. Phishing attacks were simulated by connecting to the real web site but changing the browser's address bar to display a different hostname (indicating that the web page was from an unusual source).

The toolbar study showed that attacks using a similar hostname (e.g., www.amazon-department.com to spoof www.amazon.com) have the highest spoof rate, compared with attacks using an IP address or a totally different hostname. In this study, all the attacks displayed a URL in the address bar with a hostname similar to the legitimate site. All the attacks did not use an SSL connection. Six out of the 20 tested web sites in the real world do not use SSL to protect their login pages.

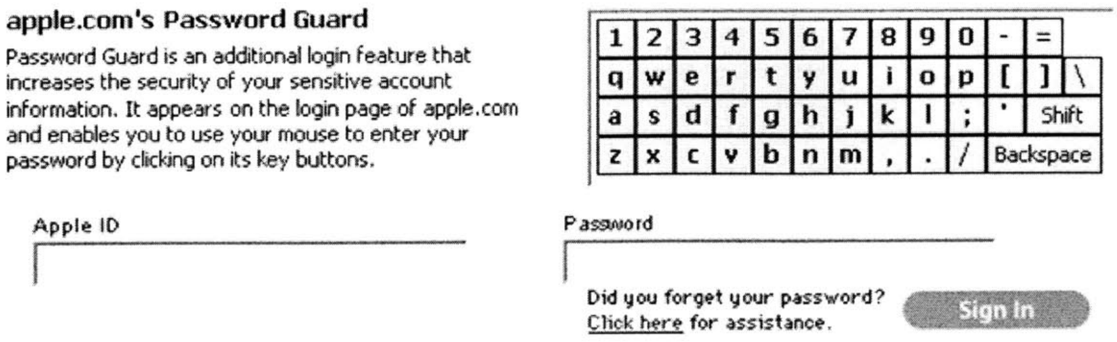
7.1.1 Simulated phishing attacks

Among the five attacks, one attack represents a normal phishing attack. In this attack, the phishing site, either copying the legitimate site or acting as a man-in-the-middle between the user and the real site, uses the same HTML login form as the one in the legitimate site. Therefore, the login form can be detected and disabled by Web Wallet. The user has to open Web Wallet to login to the phishing site.

Any new security interface should also be tested with new potential attacks. The other four attacks are designed to specifically target the Web Wallet interface.

- *Undetected-form attack*: Web Wallet analyzes the HTML source code of the web page to detect sensitive inputs, as discussed in chapter 6. It is possible that a phishing site manages to bypass the Web Wallet detection. In this attack, the login inputs are not disabled. Note, however, that any typing at the undetected inputs still makes the negative visual cue appear – e.g., the password characters zoom out of the screen.

- *Online-keyboard attack*: This is a modified undetected-form attack. In this attack, Web Wallet still fails to detect the login form. Furthermore, in order to avoid character flying when users type into web pages, the site tells the user that an online keyboard is used as an extra protection to the user's password (figure 7-1). A user is required to click the keyboard image to input his password.


apple.com's Password Guard

Password Guard is an additional login feature that increases the security of your sensitive account information. It appears on the login page of apple.com and enables you to use your mouse to enter your password by clicking on its key buttons.

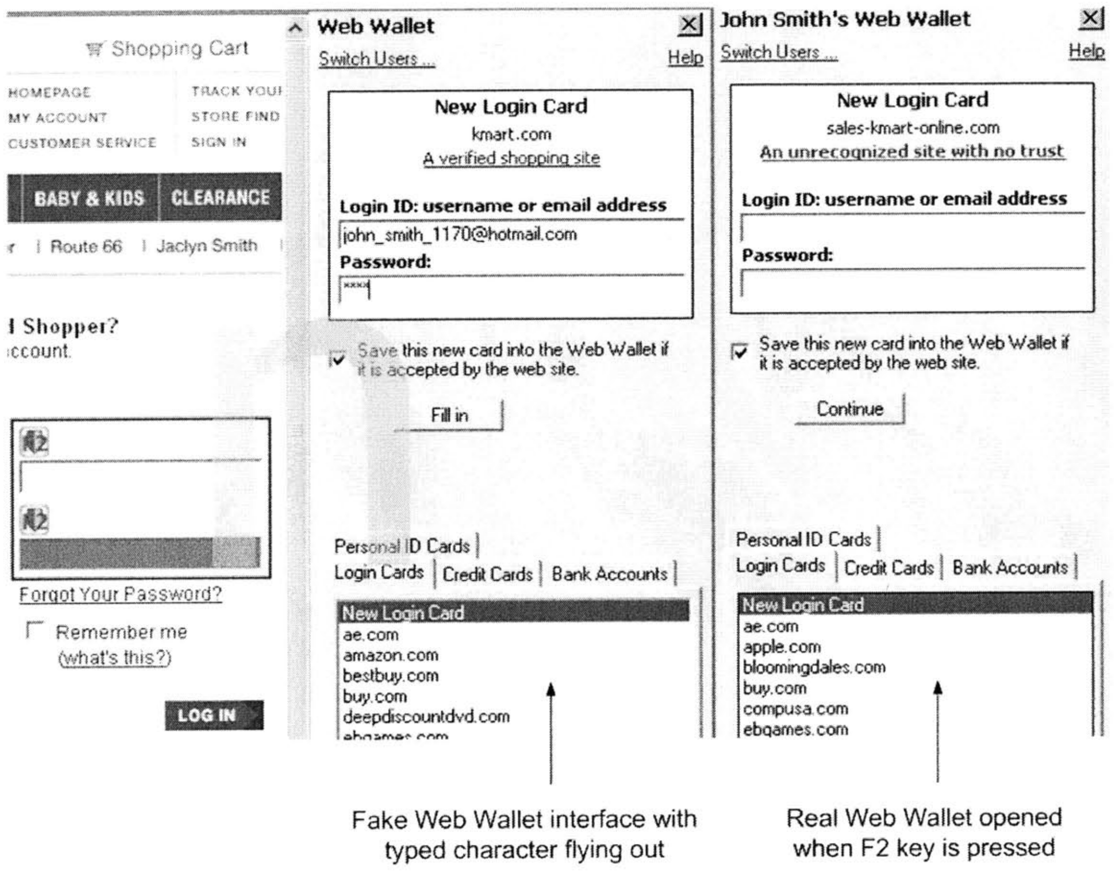
1 2 3 4 5 6 7 8 9 0 - =
q w e r t y u i o p [ ] \
a s d f g h j k l ; ' Shift
z x c v b n m , . / Backspace
Apple ID
Password
Did you forget your password?
Click here for assistance.
Sign In

Figure 7-1: Online-keyboard attack

Some legitimate sites, e.g., ING Direct, already use online virtual keyboards to evade keyboard logging attacks. [4]

- *Fake-wallet attack*: This is another modified undetected-form attack. Web Wallet fails to detect login forms. Furthermore, at a login page, a fake Web Wallet is displayed (figure 7-2). The login form is annotated in the same way as the real Web Wallet would do, but actually by the site itself. The fake Web Wallet tricks the user into typing login information at the fake sidebar. However, interaction with the fake Web Wallet produces the negative visual cue that shows the typed password character in plaintext.

- *Fake-suggestion attack*: One of the useful features of Web Wallet is its ability to suggest a safe path to the user's intended site, but this suggestion mechanism can be exploited by attackers. In this attack, the first page pops up a fake warning telling the user that the current site is a known fraudulent site and asking him to choose his intended site from a list (figure 7-3). The list, controlled by the phishing site, includes a phishing site that uses a similar name to the user's intended site and is marked as trusted. When the user chooses the phishing site from the list, the leading site performs an undetected-form attack. (The phishing site could perform any of the four attacks listed above, but the undetected-form attack was simply used in this study.)



New Login Card
kmart.com
A verified shopping site

Login ID: username or email address
john_smith_1170@hotmail.com
Password:
****

✓ Save this new card into the Web Wallet if it is accepted by the web site.

New Login Card
ae.com
amazon.com
bestbuy.com
buy.com
deepdiscountdvd.com

New Login Card
sales-kmart-online.com
An unrecognized site with no trust

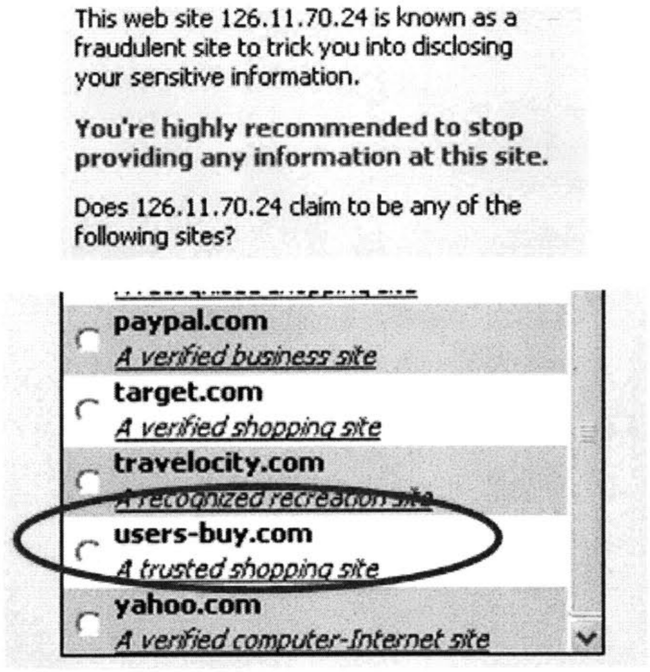
Login ID: username or email address

Password:

✓ Save this new card into the Web Wallet if it is accepted by the web site.

New Login Card
ae.com
apple.com
bloomingdales.com
buy.com
compusa.com
ebgames.com

Figure 7-2: Fake-wallet attack



This web site 126.11.70.24 is known as a fraudulent site to trick you into disclosing your sensitive information.

You're highly recommended to stop providing any information at this site.

Does 126.11.70.24 claim to be any of the following sites?

- **paypal.com** *A verified business site*
- **target.com** *A verified shopping site*
- **travelocity.com** *A recognized recreation site*
- **users-buy.com** *A trusted shopping site*
- **yahoo.com** *A verified computer-Internet site*

Figure 7-3: Fake-suggestion attack

In this study, Web Wallet rated all the phishing sites as untrustworthy and all the legitimate sites as trustworthy.

The five attacks were randomly assigned to five fixed positions: the 5th, 8th, 12th, 16th, and 19th forwarded email. Under the phishing attacks, the user may or may not have an appropriate stored login card in Web Wallet. Therefore, in this study, for each subject, two to three attacks were randomly chosen to include the stored login cards for the spoofed legitimate sites. The remaining attacks had no stored card, so the user had to type in the username and password at the Web Wallet's form panel. A total of ten login cards were initially saved and each subject had to type in login data for the other ten sites.

7.1.2 Control group

To evaluate Web Wallet, a control group was included with subjects who used Internet Explorer 6.0 (IE) without Web Wallet in it. The only security indicators for this group are the browser's address bar and the status bar. The control group saw five normal attacks with similar hostnames, presenting perfect web content but a changed URL. None of these attacks used SSL.

7.2 Study participants and protocol

A total of 21 subjects with previous experience in online shopping, 11 females and 10 males, were recruited at MIT. Thirteen subjects (62%) were college students from 11 different majors. All subjects had at least a college education. The average age was 24 (the range, 19 to 34). Fourteen subjects were randomly assigned to use Web Wallet and the other 7 were in the control group.

In order to gauge subjects' experience with online shopping, they were asked which of the 20 selected e-commerce sites they had visited. All 21 subjects had used Amazon and Yahoo, and 15 or more had used Apple, Target, Travelocity and Bestbuy. On average, each subject had used 9 sites in this study.

Before the study, subjects were briefed about the scenario and their role as John Smith's assistant. The subjects were told to be careful with John Smith's account information during the study.

I personally observed the subjects' browsing behaviors during the study. I did not interrupt the study except when subjects clicked the "report fraud" button, at which point I asked them to explain why they reported fraud and told them to stop the task at the current phishing site. At the end of the study, I interviewed the subjects by going over the unrecognized attacks to find out why they did not recognize them.

7.3 Result and discussions

Spoof rate is again the fraction of simulated attacks that successfully obtain John Smith's username and password without raising the subject's suspicion. Figure 7-4 shows the spoof rate of the normal attack with and without Web Wallet protection, and the spoof rate of all the attacks in the Web Wallet study. The Web Wallet

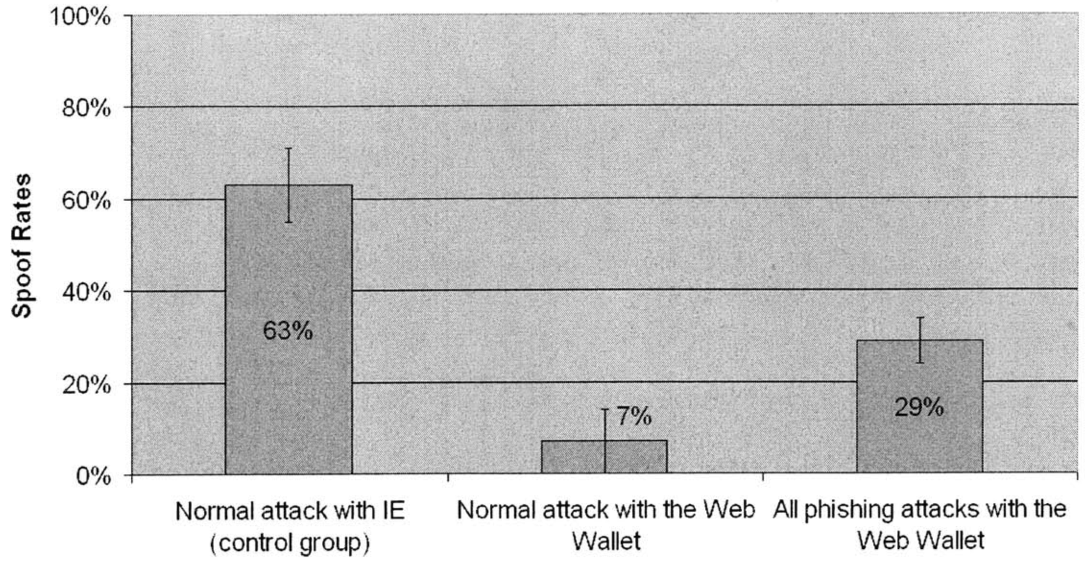


Figure 7-4: Spoof rates with and without the Web Wallet protection

protection significantly lowers the spoof rate of the normal attack from 63% to 7% (one-tail t(42) = 5.09, p < 1e-05).

Of the seven subjects in the control group, two of them reported all the phishing attacks based on (1) the odd URLs and (2) the fact that the login page is not SSL-protected. Note that one subject believed that four good sites were attacks because of the lack of SSL. The other five subjects were tricked by at least three attacks, including three subjects who were tricked by all of them, by either not looking at the URLs or explaining away the odd URLs ("the URL 'signin.travelocity.com.zaga-zaga.us' starts with 'signin' and I do want to sign in and it ends with 'us' and I know that Travelocity is in the US.").

For the rest of this chapter, I will focus on the Web Wallet test. Figure 7-5 shows the spoof rates of all the five attacks listed in section 7.1.1.

Among the 14 subjects who used Web Wallet, the first eight subjects and the last six ones used two different interfaces. I first introduce the results of the first interface, then explain why the interface was changed in the middle of the study, and finally introduce the results of the modified interface. As shown later in figure 7-6 and

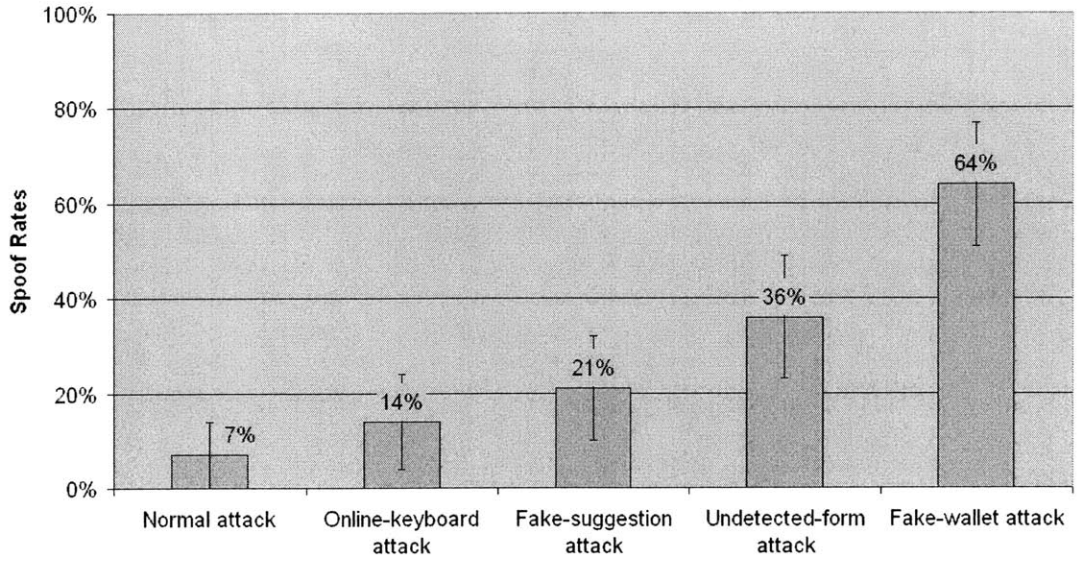


Figure 7-5: Spoof rates of the five Web Wallet attacks

figure 7-8, these two interfaces did not change the spoof rates much. Therefore, the results from these two interfaces were combined together in figure 7-4 and figure 7-5. Changing the interface did improve the security when Web Wallet was in use, however.

7.3.1 Result of the first interface

There were 40 attacks experienced by the first eight subjects. Twelve attacks were reported without Web Wallet being open (figure 7-6(a)). Six of them were detected because of the odd URLs. The other six were correctly reported as fraudulent, but the subjects were actually tricked by the fake interfaces under the attacks. In particular, five attacks were fake-suggestion attacks that were reported because (in the words of one subject) "A window popped letting me know that the website was using fraudulent methods to conceal its identity from me." But this warning window was itself fraudulent. The other one attack was the fake-wallet attack and was reported: "Not sure how to resolve the disagreement between the Web Wallet UI which reports 'radioshack.com' and the address bar of IE which reports 'radioshack.no-ip.info'. This seems suspicious." Again, this subject did not seem to suspect that the Web Wallet interface itself was fake.

Web Wallet helped to detect 17 attacks (figure 7-6(b)). Fourteen attacks were reported because the site name displayed in Web Wallet did not match the user's intention (for example, "the site name is different from compusa.com") and the site was described as untrustworthy. The other three attacks were detected when the subjects chose a stored card for login. Web Wallet warned the subjects about the discrepancy between the subject's intended site and the current site.

Eleven out of 40 attacks succeeded (figure 7-6(c)). In six attacks, the subjects failed to open Web Wallet. Four of them were the fake-wallet attacks when the subjects logged in using the fake Web Wallet. There was one subject who was tricked by all five attacks. In the interview after the study, she said that she did not trust Web Wallet and she tried to ignore everything (including the real Web Wallet warnings) except the web page content because she just wanted to get the job done. "If something happens, it is John's fault because it is he who has forwarded these emails to me."

Five successful attacks happened even when Web Wallet had been opened during the attacks. These five attacks were carefully analyzed. Two of them happened because the Web Wallet confirmation interface was originally implemented in a wrong way (figure 7-7(a)). The site list did not include the current site and acted as a generic "are you sure?" warning. The subject who got tricked bypassed the site list simply by answering no to the question "does this site claim to be any of the following sites that you have had an account with?" without even looking at the list.

Two other successful attacks reflected another problem that the Web Wallet submission was not always the path of least resistance. The standard login form was not immediately available to the subject when Web Wallet did not detect any login inputs. The subjects had to click the "new login card" entry in the card folder to open the standard form. Both attacks were undetected-form attacks, one of which was redirected by a fake-suggestion attack. Both subjects opened Web Wallet and saw the site description showing the site was untrustworthy, but they did not believe

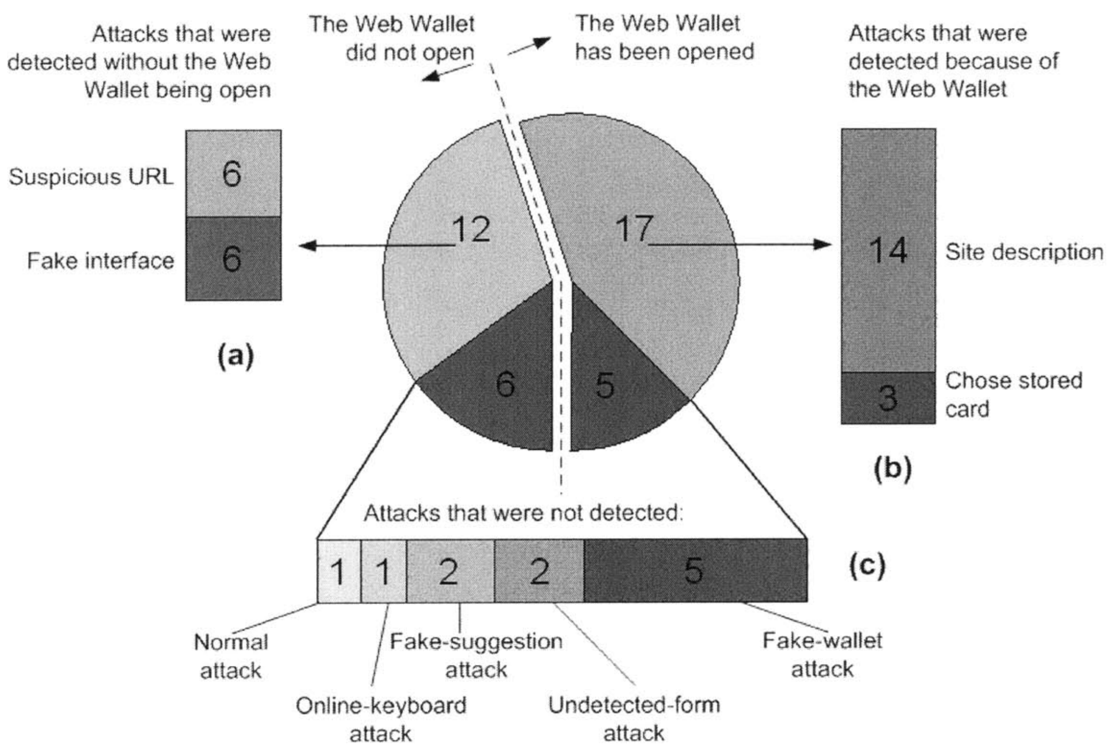


Figure 7-6: Forty attacks with the first interface

it: "The red label made me a little nervous. But everything else looked good. So I thought that it is simply the browser did not like this site. Or maybe John Smith has never done this site before." And they wanted to continue. But only the web form provided a way for direct typing, so they tried the web form ("Let's see if it works.") and got tricked.

Even though there were problems in the first interface, some value was found in the results. By blocking sensitive inputs from legitimate sites, Web Wallet could train the subjects to depend on it to login. As a result, the subjects would try Web Wallet first before they tried other newly proposed login schemes. This explained why the online-keyboard attack had the second lowest spoof rate. Under this type of attack, the subjects tended to open Web Wallet to login and eventually found the site was suspicious.

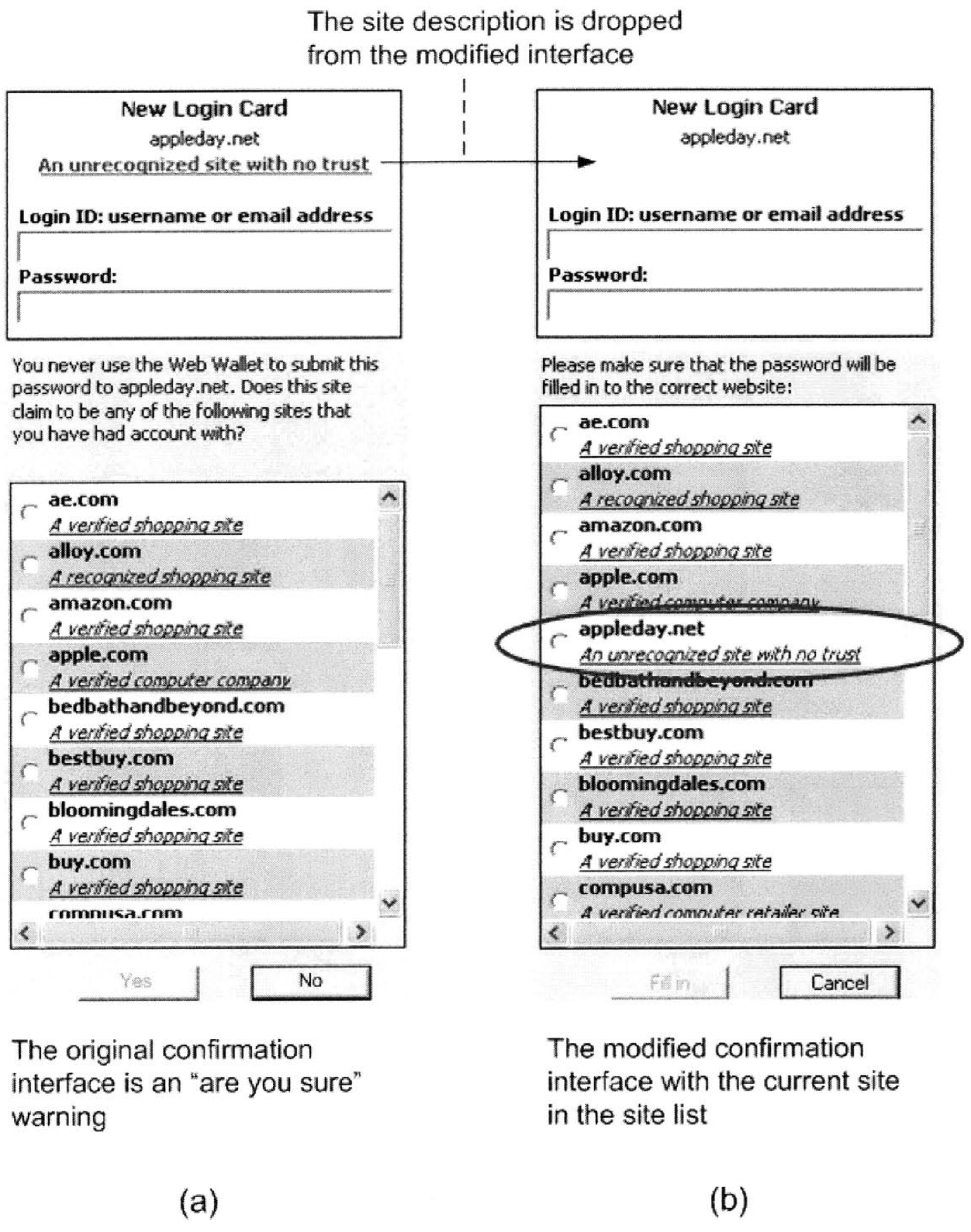
Because of the problems of the first interface, the Web Wallet interface was modified. First, the current site was added to the site list so that under phishing attacks the user had to confirm his intended site in order to continue (figure 7-7(b)).

Second, when there is no detected login input, Web Wallet displays a standard login form that supports direct typing. This improvement was intended to prevent the subjects from using the undetected login form under phishing attacks while Web Wallet is open.

The above two modifications were fixes to the problems of the interface. But the following modification is not. The site name and description displayed in Web Wallet, which act as a traditional security indicator, helped users to correctly detect 14 out of 22 attacks when Web Wallet was open. I had to deliberately drop the site description from the interface (figure 7-7) but only in this study in order to fully test other security features, like the site list and the warning based on the subject's intention.

7.3.2 Results of the modified interface

There were 30 attacks experienced by the last six subjects with the modified interface. Four attacks were detected without Web Wallet being opened (figure 7-8(a)), includ-


The site description is dropped from the modified interface
The original confirmation interface is an "are you sure" warning
The modified confirmation interface with the current site in the site list
(a)
(b)
Yes
No
Fill in
Cancel

New Login Card
appleday.net
An unrecognized site with no trust

Login ID: username or email address

Password:

You never use the Web Wallet to submit this password to appleday.net. Does this site claim to be any of the following sites that you have had account with?

ae.com
A verified shopping site
alloy.com
A recognized shopping site
amazon.com
A verified shopping site
apple.com
A verified computer company
bedbathandbeyond.com
A verified shopping site
bestbuy.com
A verified shopping site
bloomingdales.com
A verified shopping site
buy.com
A verified shopping site
compusa.com

New Login Card
appleday.net

Login ID: username or email address

Password:

Please make sure that the password will be filled in to the correct website:

ae.com
A verified shopping site
alloy.com
A recognized shopping site
amazon.com
A verified shopping site
apple.com
A verified computer company
appleday.net
An unrecognized site with no trust
bedbathandbeyond.com
A verified shopping site
bestbuy.com
A verified shopping site
bloomingdales.com
A verified shopping site
buy.com
A verified shopping site
compusa.com
A verified computer retailer site

Figure 7-7: The interface of the Web Wallet prototype modified using the study

ing one being reported because of the odd URL and the other three being reported because of the fake warning window under the fake-suggestion attacks.

In 18 attacks Web Wallet had been opened, and it helped to detect 17 of them (figure 7-8(b)). The modified interface dramatically decreased the Web Wallet failure rate from 23% (5 out of 22 attacks) to 6% (1 out of 18 attacks).

In 14 out of the 17 detected attacks, the subjects saw the site list including both the phishing site and the spoofed legitimate site. In eight of these instances, the subjects chose the legitimate site, which is their intended site, and then saw the warning telling them that the current site was fraudulent. In five instances, the subjects clicked the phishing site's description to open the site report and reported fraud. In the final instance, the subject immediately reported fraud when he found that both the phishing site and the spoofed site were in the list. Two attacks were detected when the subjects chose a stored card of their intended site and were warned that the current site was not their intended site.

Nine out of 30 attacks successfully tricked the subjects (figure 7-8(c)). But only one attack happened when Web Wallet was open. It was an undetected-form attack to spoof amazon.com. The subject opened Web Wallet at the login page and chose the stored Amazon card. She saw the warning claiming that the current site amazon-department.com was spoofing amazon.com. Instead of dealing with the warning, she then interacted with the web form. In the interview, the subject claimed that she did notice the warning but failed to pay attention to it because she felt so comfortable with the Amazon site. This problem can be easily solved: whenever a warning is displayed, Web Wallet already knows that the user's intended site is not the current site, so interaction with the current site in main browsing window should be totally blocked until the user has acknowledged the warning. As a result, in the final implementation, a modal dialog is used for site confirmation.

The following lessons were learned with the modified interface:

- Always providing affordance for typing was effective at preventing the undetected-form attack with Web Wallet being open. Six times the subjects saw an undetected form and opened Web Wallet. When facing two login forms that support

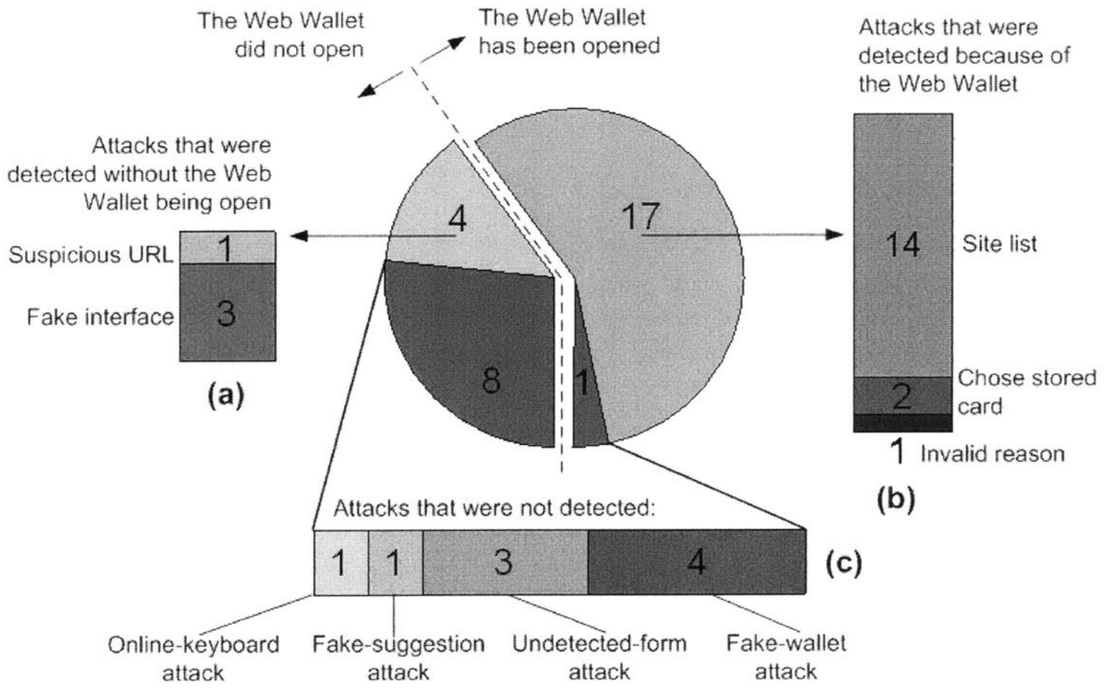


Figure 7-8: Thirty attacks with the modified interface

typing, the subjects always to type into Web Wallet because they had learned to depend on it to login and their attention was already switched to it.

- Including both the phishing site and the spoofed legitimate site in the site list helped subjects either to choose their intended site or to realize that the current site was not their intended site. It successfully prevented 14 out of 14 attacks in which the subjects saw the site list.

- The warning based on the discrepancy between the subject's intended site and the current site was effective at preventing phishing attacks because it is the fundamental danger of phishing. Combined with the results from the first interface, the subjects saw this warning 14 times. Only once did the subject fail to pay attention to it and I already discussed how to deal with it. Four times the subjects followed the suggested safe path. The other times, the subjects reported fraud based on the warning with a typical reasoning like "users-buy.com is not related to buy.com."

7.3.3 Results by attack types

Web Wallet effectively prevented the normal phishing attack. Subjects had to use Web Wallet in this attack, and there were many opportunities for them to detect the attack, including the site description, the stored card, the site confirmation interface, and the warning based on the discrepancy between their intended site and the current site.

Web Wallet effectively prevented the online-keyboard attack. As long as the subjects depended on Web Wallet to login, they used Web Wallet first before trying other proposed login schemes.

The fake-suggestion attack is risky for the attacker. Many subjects (eight out of 14) stopped at the first warning window. In the real world, the fake warning may also make users reconsider the requesting phishing email or try to type in their expected URL directly. However, if the users are successfully redirected, they tend to trust the phishing site that they are redirected to and are likely to be spoofed. Even though the real Web Wallet warns the users at the redirected phishing site, they would question Web Wallet: "you have redirected me here but why are you warning me again?"

Web Wallet failed to effectively prevent the undetected-form attack. Adding support for typing did decrease the spoof rate as long as Web Wallet was open. But because users are used to web form submission, they have a strong tendency to use it. Changing this habit is not easy. That is why it is important for Web Wallet to correctly detect and disable sensitive inputs at most legitimate sites. In that situation, Web Wallet will become the most natural sensitive data submission mechanism.

The fake-wallet attack is a very successful attack. Nine out of 14 subjects used the fake Web Wallet to login and the other five subjects reported fraud for reasons that were all unrelated to the real Web Wallet protection. Only two subjects pressed F2 to open the real Web Wallet. When he saw two Web Wallets side-by-side, one subject thought that another identical Web Wallet was open because he hit F2 by accident and thus he closed the real one and used the fake one. Most users had a wrong understanding about pressing F2. Instead of pressing F2 to secure their sensitive submission, they thought to press F2 to open Web Wallet. And since the (fake) Web Wallet is already open, they did not bother to press F2 again.

The high spoof rates of the undetected-form attack and the fake-wallet attack indicate that the negative visual cue fails. Moreover, many subjects disliked the negative visual cue. Nine of them said it was annoying or distracting. Three subjects noticed that sometimes the password was reflected in plaintext but they never thought about it seriously. Only two subjects found the visual feedback to be valuable. One reason is that beside zooming out the typed characters, another visual cue was introduced when users click the mouse at a web page (figure 7-9) in order to prevent the online-keyboard attack: whenever a user clicks on a web page, a warning icon flies up from the clicking position to top of the browsing window, where a reminder bar is located and says that "your keyboard typing and mouse clicks are going directly to the web site." As the observer in the study, I also felt that the mouse-click feedback was annoying because mouse-clicking is the main action during the browsing. Fortunately, the online-keyboard attack did not seem effective anyway, so the mouse-click feedback can be dropped. A better-designed negative visual cue on keyboard typing is a problem of the future work, as will be discussed in section 9.2.2.

Population sampling bias also exists in this study because the subjects were again recruited at MIT. Less technically savvy users may perform worse in this study. For example, if users do not know about phishing and are not motivated to use Web Wallet to protect their sensitive information, they will be more vulnerable to the undetected-form attacks. They will continue to use web forms for sensitive information submission because it is the submission mechanism they have already got used to and they do not bother to change their behavior. Moreover, if users do not realize the fact that any user interface can be spoofed, they will be more vulnerable to the fake-wallet attacks and the fake-suggestion attacks, and in turn trust less of the real Web Wallet.

On the other hand, if users are properly motivated to fully depend on Web Wallet by saving their sensitive information into Web Wallet and reuse the stored information later, Web Wallet can protect them well because Web Wallet is effective in preventing all types of tested attacks as long as it is in use. It warns users about phishing based

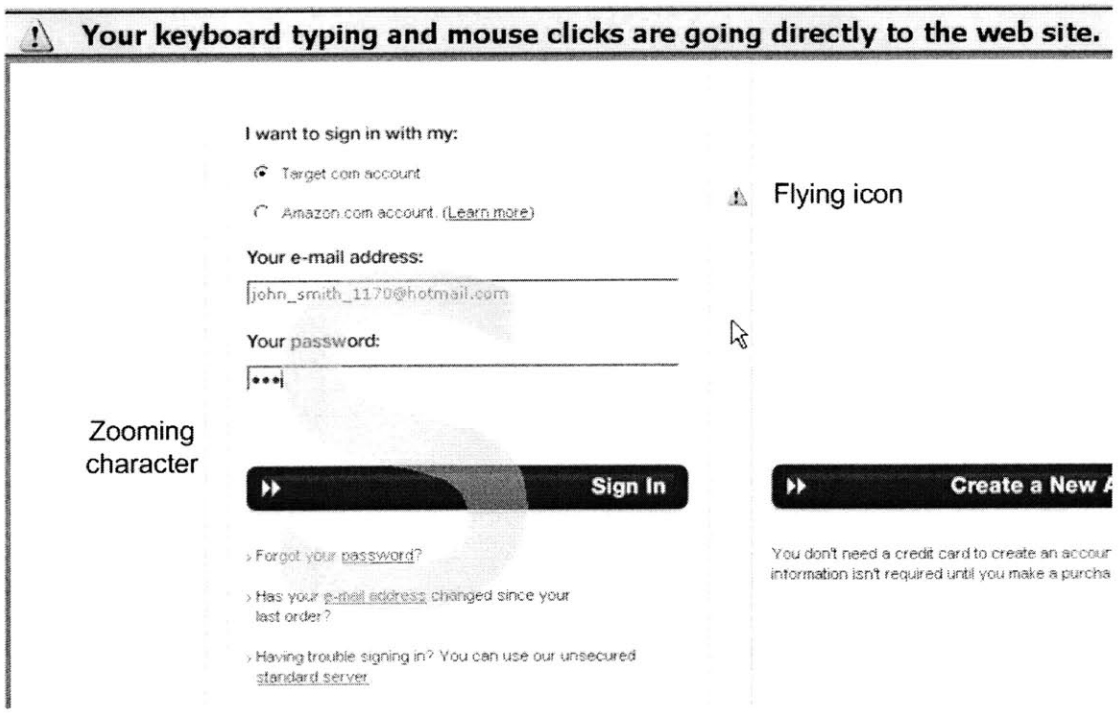

Your keyboard typing and mouse clicks are going directly to the web site.
I want to sign in with my:
Target.com account
Amazon.com account. (Learn more)
Your e-mail address:
john_smith_1170@hotmail.com
Your password:
Flying icon
Zooming character
Sign In
Create a New A
Forgot your password?
Has your e-mail address changed since your last order?
Having trouble signing in? You can use our unsecured standard server.
You don't need a credit card to create an accour information isn't required until you make a purcha

Figure 7-9: Two types of negative visual cue introduced in the study

on the user's real intention. Moreover, Web Wallet is not only an anti-phishing solution but also a secure personal information manager. It increases the security by improving usability, so that users are likely to use it because using its sensitive submission with the stored information is easier than typing data into web forms.

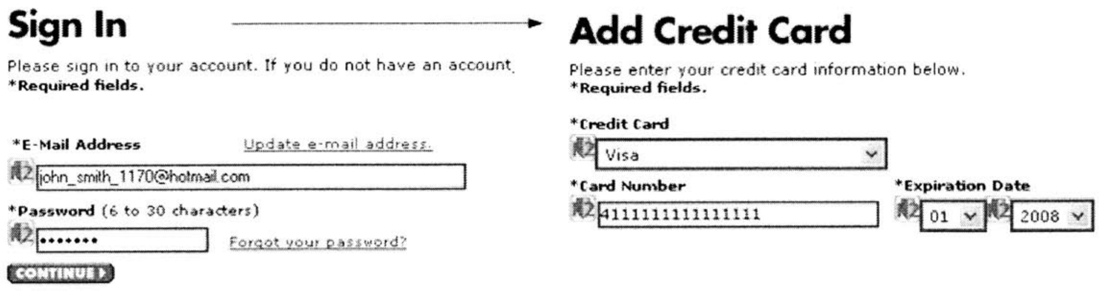
Chapter 8

Web Wallet Usability Evaluation

Chapter 7 shows that users can be successfully trained at legitimate web sites to depend on Web Wallet to submit sensitive information and thus use Web Wallet to detect the phishing sites that request sensitive information. However, the legitimate sites tested in the study in chapter 7 were optimized for Web Wallet: there are no detection errors in protected areas; there is at most one protected area in a single page, and that area, if it exists, is the one that users need to deal with; there is only one account for John Smith that users need to deal with.

This chapter concentrates on the usability aspect of Web Wallet as a personal information manager. A user study is performed to find out if users can still correctly use Web Wallet when the following complicated cases happen:

- There are some detection errors in the web page, as discussed in 5.2.3.
- There is more than one protected area in a single page. The protected area that users have to deal with is not active by default. Users have to activate that protected area and then Web Wallet can fill it.
- There is more than one account stored in Web Wallet.


Sign In

Please sign in to your account. If you do not have an account.
*Required fields.

*E-Mail Address | Update e-mail address.
john_smith_1170@hotmail.com

*Password (6 to 30 characters)
•••••••
Forgot your password?

Add Credit Card

Please enter your credit card information below.
*Required fields.

*Credit Card
Visa

*Card Number
4111111111111111

*Expiration Date
01 | 2008

Figure 8-1: A typical task: login as John Smith and add a Visa card to his account

8.1 Study design

In this study, each subject was again told to act as the personal assistant of John Smith. Because this study is focused on usability, however, there is no attack. Instead of clicking a link in a forwarded email, the subjects typed in the given URL of a web site in the browser's address bar to go to that site.

For each task, the subject either logged in as John Smith to a site or registered Alice Smith (John Smith's wife) as a new user at a site. The subject then added a credit card to either John's account or Alice's newly-created account (figure 8-1). A dummy Visa card number (4111-1111-1111-1111) was used as a valid credit card number at the tested sites.

Five detection errors (figure 5-8) were simulated in the study. Note that Web Wallet actually detects all the inputs correctly in the tested sites. The detection errors were introduced artificially by modifying the detected protected areas.

For two web sites, Web Wallet stored both John Smith's and Alice Smith's accounts. Alice Smith's login cards for both sites had the latest access time so they were automatically selected when subjects accessed these two sites. But subjects were asked to use John Smith's account to login, requiring them to notice that the wrong card was selected by default and choose the right one instead.

Compusa.com was used to find out how subjects dealt with more than one protected area in a single page. It displays both a login form and a registration form to create new users in a single page. The protected area in the login form is active by default because its protected inputs appear first in the HTML source code. One task in the study asked subjects to register Alice Smith as a new user at compusa.com. They had to switch to the inactive protected area in the registration form in order to create a new account.

Amazon.com was used to find out how subjects dealt with a partially-completed card. It separates user's registration process into two steps. In the first step, it asks for the user's email address as the username. In the second step, after making sure that the username has not been used by other registered accounts, it asks users to specify their login passwords. When a user submits and saves his username (email address) in the first step, the saved login card only has a username but no password. The login card will be completed at the second step. In order to make this registration more complicated, Web Wallet failed to detect the "email" and "confirm email" inputs in the registration form in the second step. As a result, the saved login card in the step only has a password but no username. The login card will be completed when the user logs in using that card in the next time.

Table 8.1 shows the list of tasks. The first three tasks let subjects get used to Web Wallet submission. The fourth task is designed to see if subjects can choose the intended login card from the card folder when more than one card is suitable for current login and the selected login card by default is not the intended one. The fifth task is intended to see if subjects can create a new account at amazon.com with the two registration steps and to find out how they deal with a partially saved login card. The sixth to the tenth tasks present the five detection errors, one in each task. The order of these five tasks are also randomized for each user. The eleventh task uses the new login card created in the fifth task.

In order to test Web Wallet as a personal information manager, John Smith's account and his Visa card are already stored in Web Wallet. At the beginning of the study, subjects were given a piece of paper with John Smith's and Alice Smith's personal information, e.g., address and phone number, which is necessary for subjects

Index	Website	Tasks	Testing purpose
1	ae.com	Login as John Smith and add his Visa card to his account	Get used to the Web Wallet submission
2	bestbuy.com	(same as 1)	(same as 1)
3	staples.com	(same as 1)	(same as 1)
4	deepdiscountdvd .com	Login as John Smith and clear his wishlist	Multiple accounts
5	amazon.com	Register Alice Smith as a new user	Partially saved card and multiple accounts
6	pcconnection.com	(same as 1)	Detection error 2 (section 5.2.3, figure 5-8(b))
7	macys.com	(same as 1)	Detection error 5 (section 5.2.3, figure 5-8(e))
8	compusa.com	Register Alice Smith as a new user and add John's Visa card to the new account	Detection error 3 (section 5.2.3, figure 5-8(c)) and multiple protected areas in a single page
9	walmart.com	(same as 1)	Detection error 1 (section 5.2.3, figure 5-8(a)) and multiple accounts
10	paypal.com	(same as 5)	Detection error 4 (section 5.2.3, figure 5-8(d))
11	amazon.com	Login as Alice Smith using the newly-created account	

Table 8.1: The list of tasks in the user study

to register Alice Smith as a new user and to add a credit card to John Smith's account.

8.2 Results and discussion

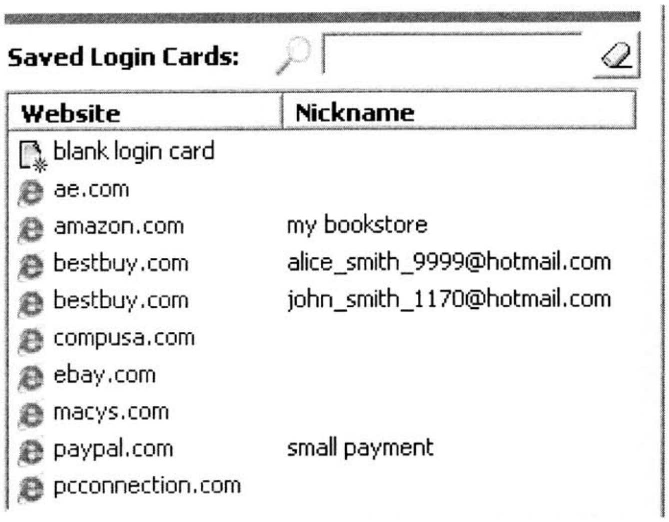
Six MIT students, three males and three females, with the average age of 23 (range from 21 to 24), were recruited to join this study.

With the first two subjects, John Smith's login information and credit card information were listed in the piece of paper that was given to them at the beginning. As a result, these two subjects tended to use Web Wallet purely as a submission mechanism. They typed John's information from the paper into Web Wallet and submitted it into web pages without saving it in the card folder. They did not bother to look at the stored cards in the card folder. The study scenario then changed because of this observation. John's login information and credit card information were no longer provided on paper. Instead, the remaining four subjects were told that these two pieces of information have been stored in Web Wallet. As a result, they had to deal with the stored cards in the card folder. They also saved Alice Smith's newly-created accounts into Web Wallet.

One subject did not know how to deal with multiple stored accounts. She was confused by the number at the end of the domain name. The number "2" is meant to indicate that it is the second saved account for walmart.com. But this subject thought that there were two saved accounts under this node and she did not know how to open this node to get the accounts. To solve this confusion, if there are more than one saved login cards for a specific site, the login username is assigned as the card's nickname if this card does not have a nickname (figure 8-2).

Because clicking an input element in a web page in order to give it keyboard focus is natural to users, all subjects had no difficulty clicking an inactive protected area in order to activate it.

The subjects correctly dealt with the missing protected entry at walmart.com. They clicked the link "Click here if this form is incomplete or wrong", added a card number entry, clicked the "submit" button and dragged the card number from Web



Saved Login Cards:

Website	Nickname
blank login card	
ae.com	
amazon.com	my bookstore
bestbuy.com	alice_smith_9999@hotmail.com
bestbuy.com	john_smith_1170@hotmail.com
compusa.com	
ebay.com	
macys.com	
paypal.com	small payment
pcconnection.com	

Figure 8-2: Using username as the card's nickname to differentiate multiple login accounts at bestbuy.com

Wallet to the corresponding HTML input element. Interestingly, even though the card number is displayed in the newly-added card number entry, no one typed the number into the web page. One possible reason is that dragging an icon is easier than typing a 16-digit card number.

The subjects correctly dealt with the missing protected area for credit card information at macys.com. They submitted the standard payment form containing the stored Visa information, and dragged each entry from Web Wallet to its corresponding input element in the web page. This observation shows that the main reason that the subjects used the drag-and-drop mechanism is not merely because it is easier than typing a long string. After all, they also dragged the card type information into the page even though the card type can be easily set at the web page by selecting an option in the "card type" select element. Rather, drag-and-drop is a natural means of submission. The subjects depended on Web Wallet since it stores the data that they wanted to submit and their attention has focused on the Web Wallet interface. It is natural for them to follow the Web Wallet's instructions.

The subjects successfully used Web Wallet to deal with the sensitive input that asks for the card number but is mistakenly detected as the card verification code (task

8). However, instead of right-clicking the Web Wallet icon to change the sensitive type directly, the subjects added a new card number entry and dragged it to the web page to overwrite the Web Wallet's detection. In general, it is hard for users to find the hidden context menu by right-clicking.

The subjects did not bother to correct the detection error that mistakenly treats some insensitive inputs as sensitive. A common workaround used by the subjects was submitting the data from Web Wallet to the web page and modifying the input value at the enabled inputs in the web page. The main problem with this approach is that users keep seeing the same detection error when they access the same page. But they have no difficulties dealing with this error.

The subjects had problems handling the protected area at paypal.com all of whose inputs are mistakenly protected (task 10). They first tried to add a phone number as a sensitive input, but this failed because phone numbers are not one of Web Wallet's protected types. One subject then typed the given phone number into the credit card entry in Web Wallet and submitted it. Fortunately, he only did submission, not saving the phone number as another credit card into Web Wallet. The other subjects eventually found out by trial and error that there is a way to unprotect an entry by right-clicking its Web Wallet icon, and they unprotected the home phone number input. However, they did not bother to unprotect the following two inputs for optional working phone number. Even worse, two subjects kept using right click to unprotect all the protected entries in the web page, even though some entries were correctly protected.

The Web Wallet interface can be improved in two ways to prevent users from unwisely unprotecting entries. One is to make the mapping visually obvious between the disabled protected area in the web page and its submission interface in Web Wallet. Without a strong connection, a disabled input is mainly an obstacle for data submission. On the other hand, with a strong connection, users will know that Web Wallet is the right submission interface and will use it. Links have been proposed to visualize the interconnections between different UI elements. [35] Displaying a link between a protected area and its submission interface can help users understand that

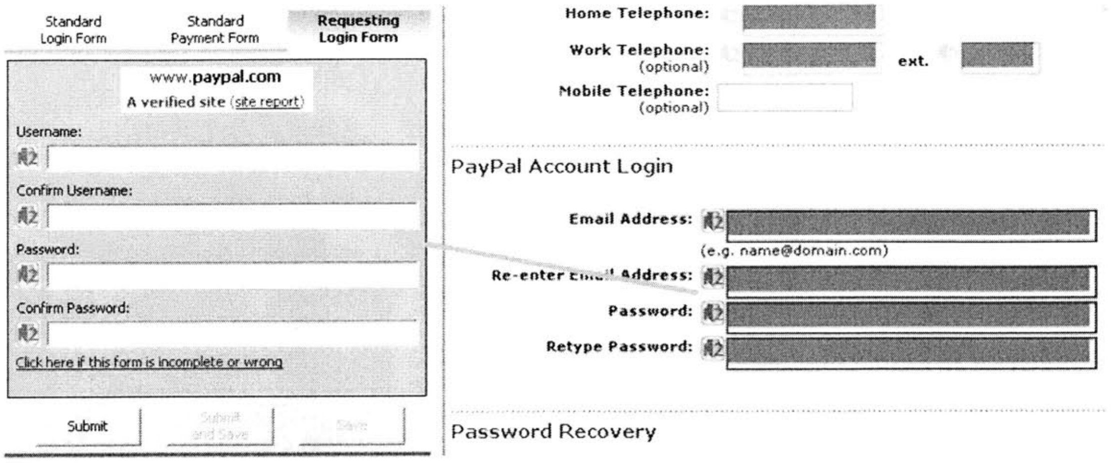
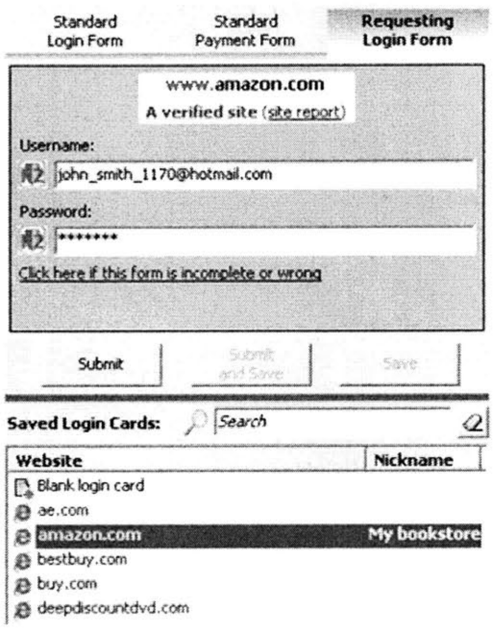


Figure 8-3: Visualize the mapping between a protected area and its submission interface at Web Wallet (simulated interface)

a *disabled* input is actually a *protected* input (figure 8-3). The second improvement is to make unprotecting undoable. One subject commented that he noticed that he was in the middle of "destroying" a correctly-protected registration form when he was about to unprotect a password entry. He wanted to undo the unprotecting process but he did not know how, so he had to continue. An undo mechanism would have helped him to correct his mistakes.

One major problem found when subjects registered Alice at amazon.com is that the card folder lacks a blank card. In this study, the card folder already has a stored John Smith's login card to amazon.com. That card was automatically selected and filled into the Web Wallet form panel when a subject was at amazon.com. With the filled form panel, no subject knew how to register Alice as a new user. They knew that the inputs in the form panel are editable but they did not want to clear the filled data because "that may overwrite John Smith's account". They did search the card folder for an appropriate card. So a blank card will help them register another account when an existing account is selected by default (figure 8-4).



www.amazon.com
A verified site (site report)

Username: john_smith_1170@hotmail.com

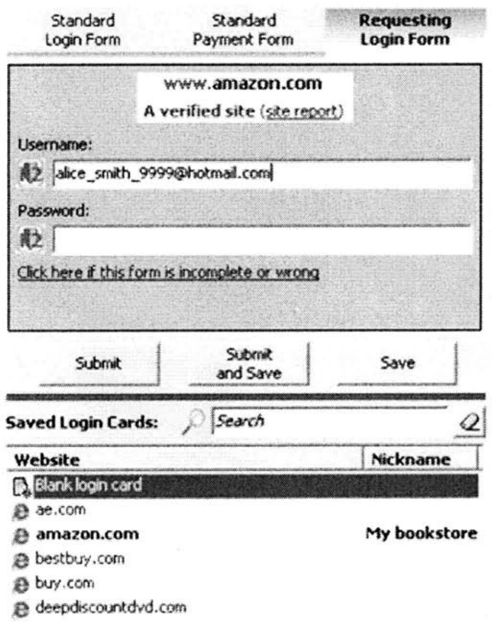
Password: *******

Click here if this form is incomplete or wrong

Saved Login Cards:

Website	Nickname
Blank login card	
ae.com	
amazon.com	My bookstore
bestbuy.com	
buy.com	
deepdiscountdvd.com	

Web Wallet automatically selects John Smith's saved amazon.com login card



www.amazon.com
A verified site (site report)

Username: alice_smith_9999@hotmail.com

Password:

Click here if this form is incomplete or wrong

Saved Login Cards:

Website	Nickname
Blank login card	
ae.com	
amazon.com	My bookstore
bestbuy.com	
buy.com	
deepdiscountdvd.com	

Use the blank login card to register Alice Smith as a new user

Figure 8-4: Add a blank card in the card folder

8.2.1 Lessons learned

The following lessons were learned from the usability study:

- A strong visual connection is important to prevent users from incorrectly unprotecting sensitive entries. A disabled HTML input can be filled and is protected by Web Wallet.
- Drag-and-drop has been proved to be a natural and usable submission mechanism for users to deal with undetected sensitive inputs.
- Users have difficulties finding and using context menus that can only be triggered by right-clicking.
- Having a blank card in the card folder is very important for users to submit and store new information to Web Wallet.
- Clicking an inactive protected area to activate it is intuitive to users because it is the way to make an element being focused on keyboard input.

Another lesson is worth explaining in more detail. Encouraging users to save sensitive data into Web Wallet improves both security and usability. Users depending on Web Wallet to submit stored data can prevent both the undetected-form attack and the fake-wallet attack, the two serious attacks found in chapter 7. Moreover, users dragging stored data into the web page can help to correct certain detection errors, as shown in this study.

However, many users do not feel comfortable storing sensitive information, especially financial information, in their personal computer. When I did the security evaluation of the Web Wallet prototype, I interviewed the subjects for their opinions about the browser's built-in password and form manager that stores and auto-fills the user's personal information online. Two biggest concerns were that the stored information may be accessed, either by a computer virus or spyware, or by other people who share the same computer.

To prevent unauthorized people from accessing the stored information, Web Wallet users can specify a master password for Web Wallet. The master password is requested the first time the card folder is accessed. The master password expires when all IE browser windows are closed. The current Web Wallet interface lacks an explicit "log out" button. Moreover it is better to expire the master password after a period of idle time. Whether a master password can encourage users to store data into Web Wallet needs further testing.

8.3 Heuristic usability comparison of Web Wallet with existing auto-fill applications

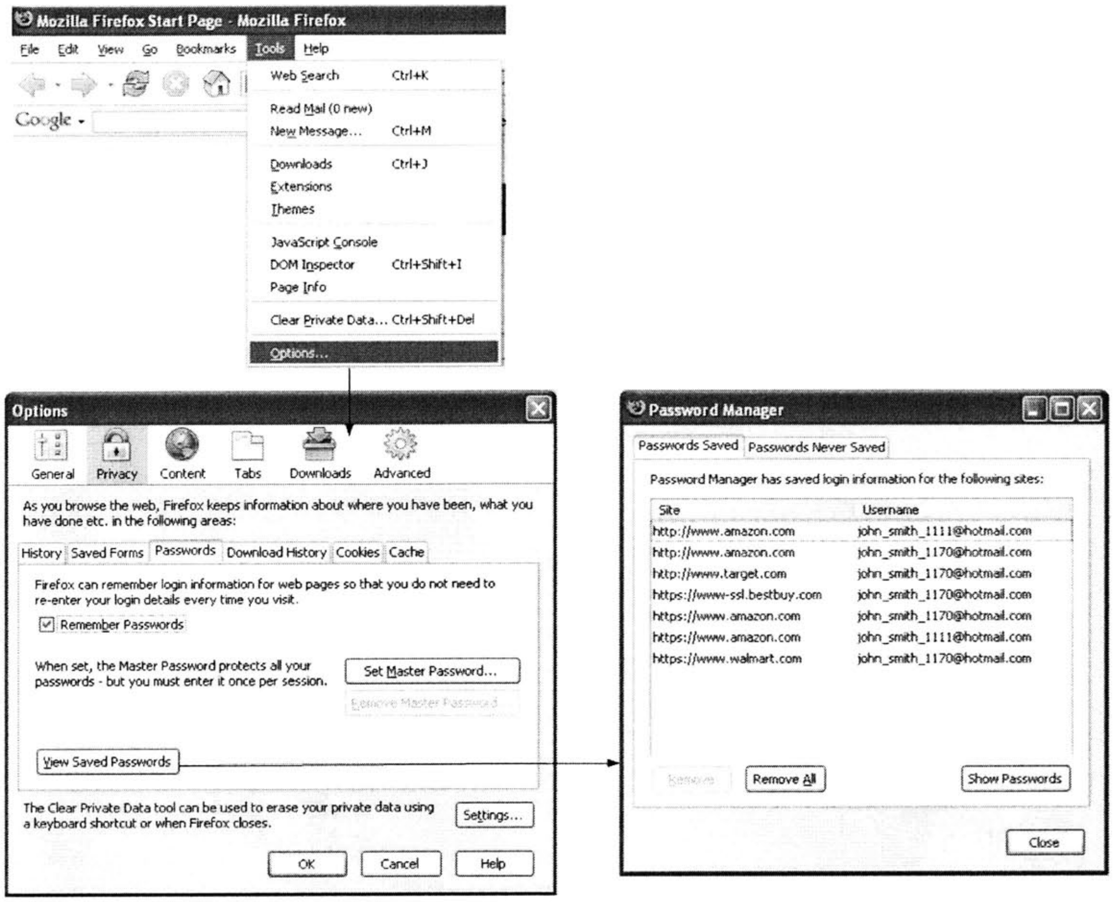
Auto-fill applications make user's online interaction easy by remembering a user's typing input at a web form the first time when he uses this form. When the user accesses the same form again, the saved data can be automatically filled in.

Existing auto-fill applications can be generally separated into two categories. The first category is the web browser's own password and form managers. The second category includes applications that are developed by third parties and integrated into the browser as plug-ins, such as Google toolbar and RoboForm. In this section, Web Wallet is compared with applications from both these categories from the usability point of view.

8.3.1 Comparison of Web Wallet with browser's password manager

In the comparison, Firefox's password manager is used as an example. The password managers from other major browsers, like Internet Explorer and Mozilla, have the same problems.

One major problem of Firefox's password manager is its lack of visibility. Users cannot immediately know at what sites they have passwords saved in the browser. They have to dig into the options dialog box to view the list of the saved passwords



As you browse the web, Firefox keeps information about where you have been, what you have done etc. in the following areas:

Firefox can remember login information for web pages so that you do not need to re-enter your login details every time you visit.

[✓] Remember Passwords

When set, the Master Password protects all your passwords - but you must enter it once per session.

The Clear Private Data tool can be used to erase your private data using a keyboard shortcut or when Firefox closes.

Password Manager has saved login information for the following sites:

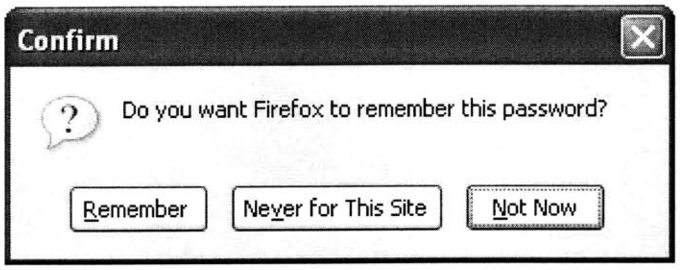
Site	Username
http://www.amazon.com	john_smith_1111@hotmail.com
http://www.amazon.com	john_smith_1170@hotmail.com
http://www.target.com	john_smith_1170@hotmail.com
https://www-ssl.bestbuy.com	john_smith_1170@hotmail.com
https://www.amazon.com	john_smith_1170@hotmail.com
https://www.amazon.com	john_smith_1111@hotmail.com
https://www.walmart.com	john_smith_1170@hotmail.com

Figure 8-5: Steps to view the saved passwords in Firefox

(figure 8-5). One possible reason for hiding the password management from users is that browser might want users not to worry about the login process once the login information is saved.

However, this lack of visibility introduces another problem, in that users lose control of the auto-fill mechanism. Users cannot decide which login account to use if they have multiple saved accounts for a single site. They cannot easily delete invalid saved login accounts. They cannot overwrite the saved account: any difference in username or password will create a new entry in the password list.

Lack of efficiency and flexibility is another usability problem of the password manager. After a user presses the submit button in order to login, it always pops up a dialog to confirm with the user to save the submitted information if it is different


Confirm

Do you want Firefox to remember this password?

Figure 8-6: Firefox's confirm dialog for saving a new password

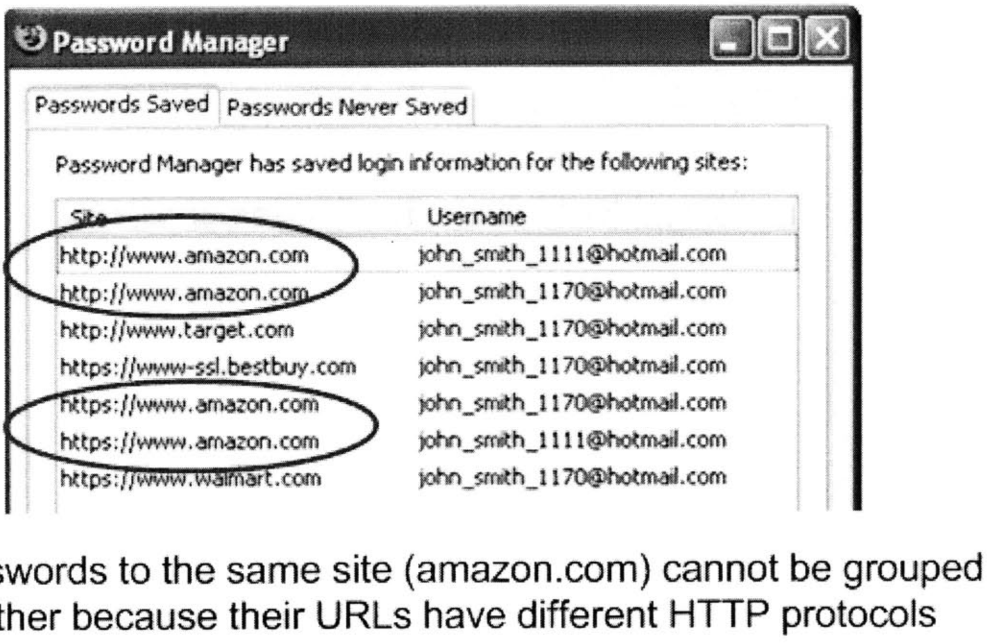
from the stored information (figure 8-6). This confirmation interrupts the user's login process. Moreover, the password manager cannot remember a user's new login account at a site's registration form. One possible reason is that the new account may be rejected by the site because either there is a conflict in the username or the password does not meet the specified requirement. The password manager does not want to save an account that will be rejected by a site. On the other hand, if users can easily delete invalid saved accounts, saving login account during registration should not be a problem. The saved password list (figure 8-7) has a usability problem too. It does not have a search feature and the sites in the list are sorted based on the whole URL. As a result, if the list is long, users will have a hard time finding all the saved passwords for a specific site.

Web Wallet does not have the usability problems mentioned above. All the saved login accounts are listed in the card folder. Users can switch to different login accounts by simply selecting them. When a user modifies a selected account, he will be asked whether to replace the selected account or to create a new account (figure 8-8).

Web Wallet has both a "submit" button and a "submit and save" button. By clicking different buttons, users explicitly indicate if they want to save the account or not. No confirmation dialog is needed anymore.

Web Wallet supports an incremental search on both the hostnames and the nicknames of the login accounts. Moreover, it groups the login accounts based on their domain names instead of the whole URLs.

Web Wallet protects all login-account related submission, including registering a



Password Manager has saved login information for the following sites:

Site	Username
http://www.amazon.com	john_smith_1111@hotmail.com
http://www.amazon.com	john_smith_1170@hotmail.com
http://www.target.com	john_smith_1170@hotmail.com
https://www-ssl.bestbuy.com	john_smith_1170@hotmail.com
https://www.amazon.com	john_smith_1170@hotmail.com
https://www.amazon.com	john_smith_1111@hotmail.com
https://www.walmart.com	john_smith_1170@hotmail.com

Passwords to the same site (amazon.com) cannot be grouped together because their URLs have different HTTP protocols

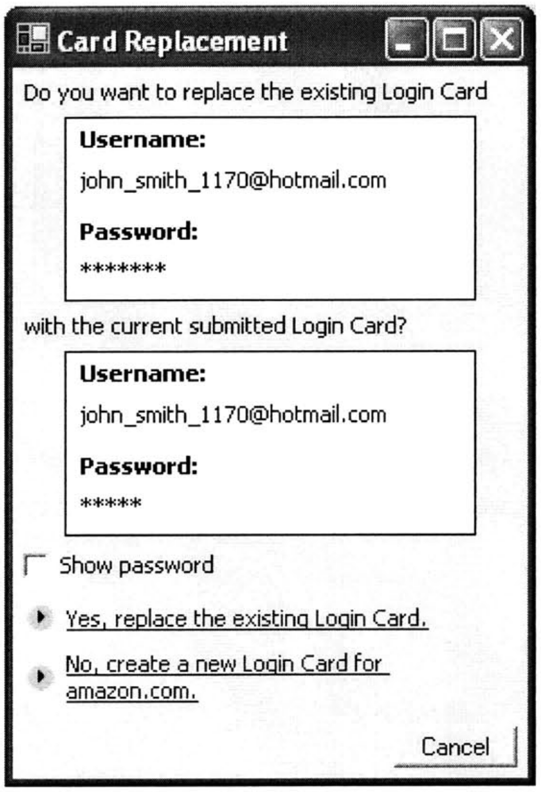
Figure 8-7: Firefox's password list are sorted based on the whole URLs of the login sites

new account and updating an existing account. Web Wallet can therefore save a new account at the registration step.

8.3.2 Comparison of Web Wallet with Google toolbar and RoboForm

Google toolbar and RoboForm analyze the current web page in order to find out if there is any HTML input that can be filled with the stored personal information. One major problem is that users cannot correct their input detection errors. For example, if a set of inputs asking for credit card information are failed to be detected, users have to manually input the credit card information every time they use the same form.

RoboForm does not visually indicate which inputs that can be filled and has been filled. It raises a security problem in that it may fill in some sensitive data without user's acknowledgement. A quick fix that it uses is to pop up a dialog box every



Card Replacement

Do you want to replace the existing Login Card

Username:
john_smith_1170@hotmail.com

Password:
****

with the current submitted Login Card?

Username:
john_smith_1170@hotmail.com

Password:
*****

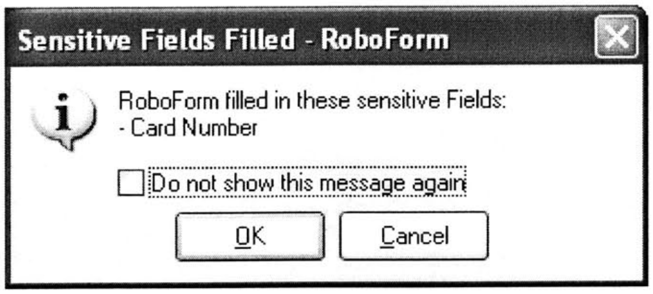
Show password

Yes, replace the existing Login Card.

No, create a new Login Card for amazon.com.

Cancel

Figure 8-8: Web Wallet's dialog for the login card replacement


Sensitive Fields Filled - RoboForm

RoboForm filled in these sensitive Fields:
- Card Number

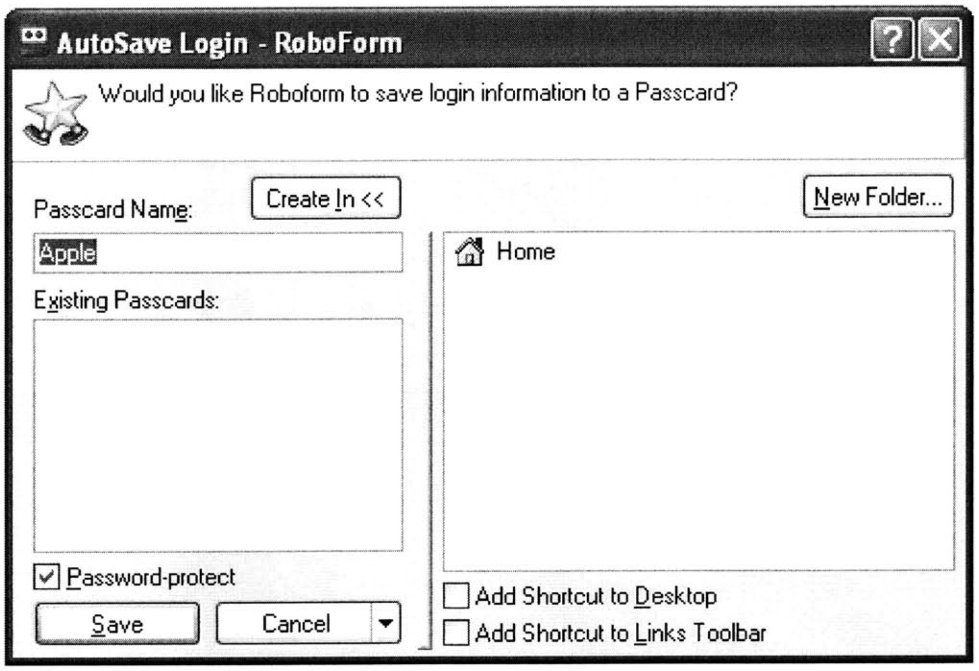
Figure 8-9: RoboForm's warning for credit card information fill-in

time credit card information is filled (figure 8-9). This confirmation again interrupts the user's submission process. RoboForm has the same efficiency problems as the browser's password manager. It confirms with the user in order to save the login information (figure 8-10) and it cannot remember a user's new account information that is typed into a registration form.

In order to use saved personal information, both Google toolbar and RoboForm require users to input the information first (figure 8-11). This is an extra configuration step that many users do not bother to do.

RoboForm claims to effectively prevent phishing by showing the passcards only if a URL stored in the passcard matches the current site's URL. [56] But if a user indicates that a passcard should be used, the information is filled in without any warning, as shown in figure 8-12. This is a serious problem for RoboForm as an anti-phishing solution.

Web Wallet does not have the usability problems mentioned above. Users can correct Web Wallet's detection errors so that when they access the same form again, the form is correctly detected. Web Wallet saves the user's sensitive information when such information is first submitted. As a result, Web Wallet does not require users to store their sensitive information beforehand. More importantly, Web Wallet effectively warns users when it detects that they are going to submit a login card to a different site from the site that the login card is supposed to go.


AutoSave Login - RoboForm

Would you like Roboform to save login information to a Passcard?

Passcard Name: Apple

Existing Passcards:

[✓] Password-protect

Home

[ ] Add Shortcut to Desktop

[ ] Add Shortcut to Links Toolbar

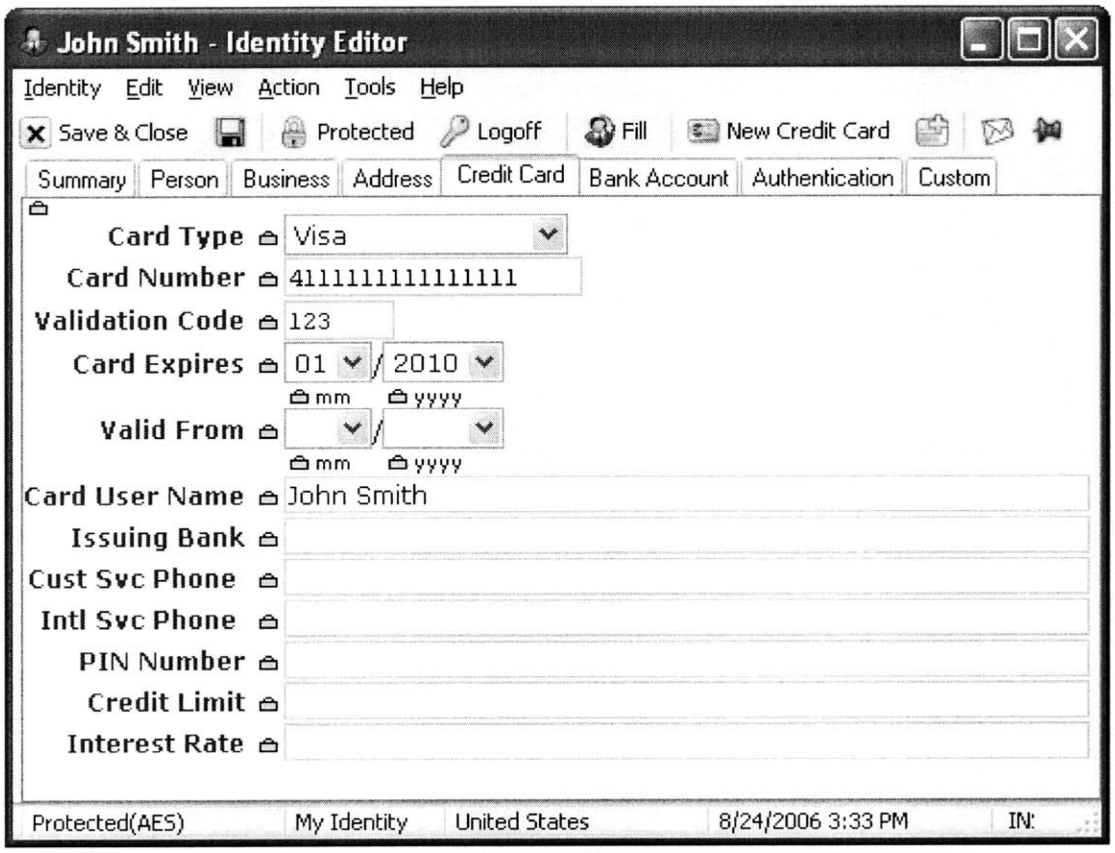
Figure 8-10: RoboForm's confirm dialog for saving new password

8.3.3 Heuristic evaluation of Web Wallet

Web Wallet has its own usability problems. As discussed before, current Web Wallet lacks a link between the disabled area in the web form and the submission interface at Web Wallet. It also lacks an undo mechanism when users make mistakes in correcting the Web Wallet's detection errors.

The Web Wallet's site confirmation dialog interrupts the user's sensitive submission at unverified sites. This feature is introduced for a security reason: users should be very clear about the connecting site before they submit their sensitive information. An improvement is that the hash value of the user's submission is remembered by Web Wallet so that the next time when the user submits the same sensitive information at the same site, no site confirmation is required. Moreover, Web Wallet does not bother to confirm the user's intended site if the current site is verified.

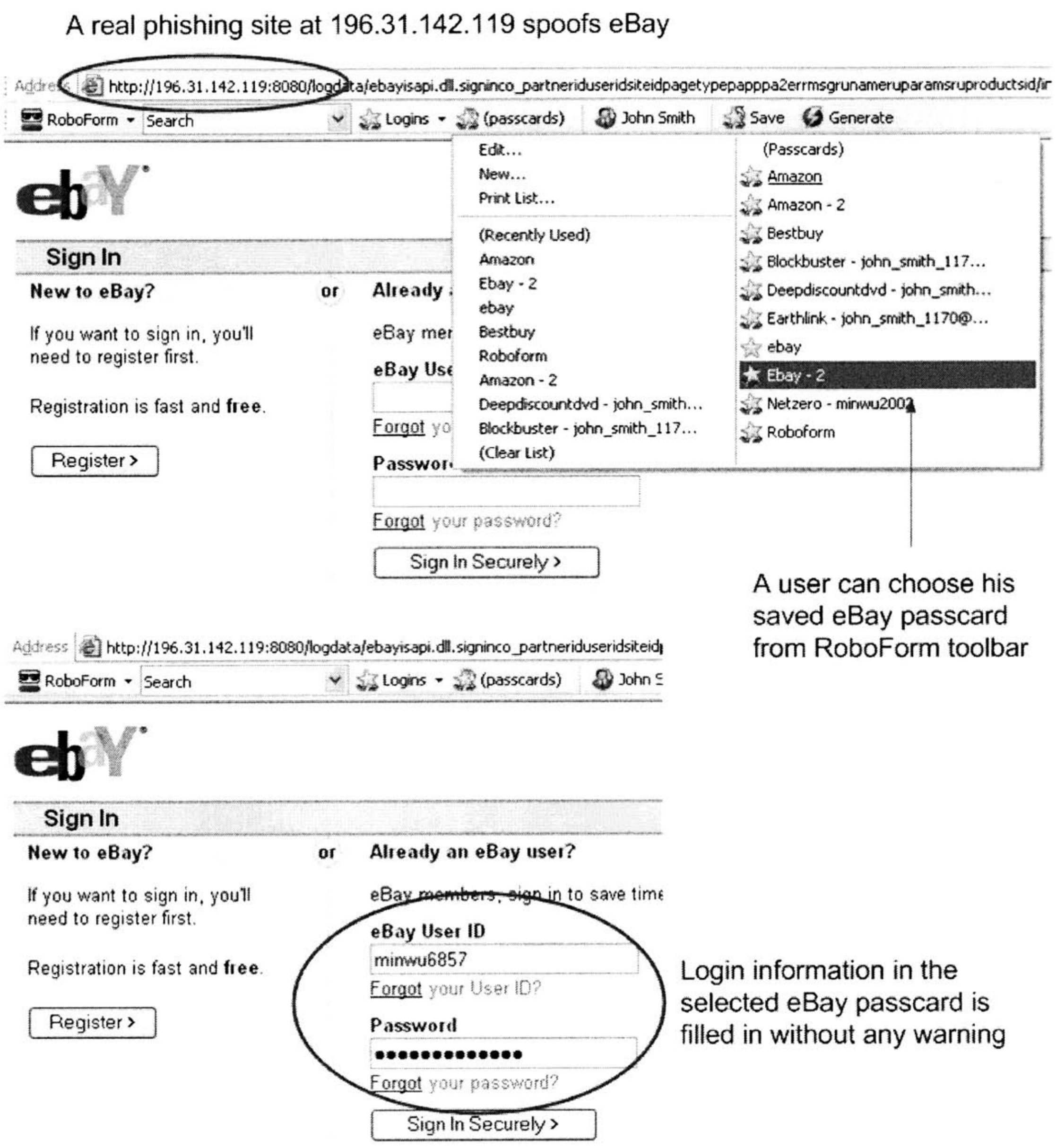
Web Wallet introduces a negative visual cue by zooming out the characters that users type in at web pages. This negative cue is introduced in order to warn users that submitting sensitive information at web forms is potentially dangerous. However,


John Smith - Identity Editor
Identity Edit View Action Tools Help
Save & Close
Protected
Logoff
Fill
New Credit Card
Summary Person Business Address Credit Card Bank Account Authentication Custom

Card Type	Visa
Card Number	4111111111111111
Validation Code	123
Card Expires	01 / 2010 (mm / yyyy)
Valid From	/ (mm / yyyy)
Card User Name	John Smith
Issuing Bank	
Cust Svc Phone	
Intl Svc Phone	
PIN Number	
Credit Limit	
Interest Rate	

Protected(AES) | My Identity | United States | 8/24/2006 3:33 PM | IN:

Figure 8-11: RoboForm's identity editor


A real phishing site at 196.31.142.119 spoofs eBay

A user can choose his saved eBay passcard from RoboForm toolbar

Login information in the selected eBay passcard is filled in without any warning

Address http://196.31.142.119:8080/logdata/ebayisapi.dll.signinco_partneriduseridsiteidpagetypepapppa2errmsgrunameruparamsruproductsid/
RoboForm Search Logins (passcards) John Smith Save Generate
Edit...
New...
Print List...
(Recently Used)
Amazon
Ebay - 2
ebay
Bestbuy
Roboform
Amazon - 2
Deepdiscountdvd - john_smith...
Blockbuster - john_smith_117...
(Clear List)
(Passcards)
Amazon
Amazon - 2
Bestbuy
Blockbuster - john_smith_117...
Deepdiscountdvd - john_smith...
Earthlink - john_smith_1170@...
ebay
Ebay - 2
Netzero - minwu2002
Roboform
Sign In
New to eBay?
or
If you want to sign in, you'll need to register first.
Registration is fast and free.
Register >
Already an eBay user?
eBay members, sign in to save time
eBay User ID
minwu6857
Forgot your User ID?
Password
Forgot your password?
Sign In Securely >

Figure 8-12: Filling a RoboForm's password at a phishing site without being warned

this negative cue needs to be better designed because most users find it distracting. Designing a better negative cue is a question for future work, as will be discussed in section 9.2.2.

Web Wallet uses a small font in order to display more information in a limited area. A browser's sidebar, when is open, should not take too much space from the main browsing window. Web Wallet may have a readability problem as a result, especially to older people.

Web Wallet displays different background colors based on the current site's reputation. A green background indicates a verified site, while red means the current site is suspicious or known fraudulent. Such a distinction may not be obvious to people with red-green color blindness. Other types of visual variables, such as texture and value, should be added to help those users to better visually differentiate verified sites from suspicious ones.

8.4 Web Wallet is a usable personal information manager

Web Wallet has been tested to be a usable personal information manager. In the study, subjects had little difficulty dealing with most of its sensitive-input detection errors. They could correctly deal with multiple accounts for a single site and multiple protected areas in a single page. Feasible solutions are given to the usability problems that were found in the study.

Web Wallet is more user-friendly compared with existing auto-fill applications. It gives users a full control to their saved information. It does not unnecessarily interrupt the user's browsing session. It does not require user to store sensitive information beforehand as an extra configuration step. Users can easily correct the Web Wallet's sensitive-input detection errors so that the Web Wallet's auto-fill mechanism can be fully and effectively used.

Chapter 9

Conclusion and Future Work

The problem that this thesis concentrates on is deceptive phishing. Deceptive phishing tricks users into leaking their sensitive information online at a fake web site that spoofs an existing legitimate web site.

The fundamental problem of deceptive phishing is that when a user submits sensitive information online under the attack, his mental model about this submission is different from the system model that actually performs this submission. Specifically, the system sends the data to a different web site from the one where the user intends to submit the data. The fundamental solution to deceptive phishing is to bridge the semantic gap between the user's mental model and the system model.

There are two major approaches to bridging the gap. One approach is to reflect the system model to the user. Anti-phishing toolbars and the browser's security indicators take this approach. The toolbar studies in chapter 3 show that this approach is not effective. Users are required to constantly pay attention to the toolbar message and to have the expertise to always correctly interpret the toolbar message. Normal users meet neither of these requirements.

The other approach is to let the user tell the system their intentions when they are submitting data online. The system can then check if the actual submission meets the user's intention or not. If not, the system can effectively warn the user about this discrepancy and provide a safe path to the user's intended site. Web Wallet takes this approach and has been proved to be an effective and promising anti-phishing solution.

9.1 Anti-phishing recommendations

Internet companies need to follow some standard practices to better distinguish their sites from malicious phishing attacks. Companies should use a single domain name that matches their brands name rather than using IP addresses or multiple domain names for servers. They should use SSL to encrypt every web page on their sites. SSL certificates should be valid and from widely used CAs. They should deal with outsourcing carefully. With outsourcing, different domains are related to a single organization. In order to make the domain name of an organization consistent, outsourcing should be discouraged. But if outsourcing is necessary, then authentic outsourcing information should be available to the web browser and anti-phishing software. Behera and Agarwal have proposed a mechanism [3] where servers provide security relevant metadata (including the outsourcing information) to the web browser via a standard protocol.

Providing a safe path to the user's intended site when the user's intended site is different from the current site has been proved to be a useful anti-phishing feature in the user study discussed in chapter 7. It should be added to phishing blacklist. Most phishing blacklists are manually maintained. Human experts examine the user's report and decide which sites should be included into the list. At the same time, the legitimate site that spoofed by the phishing attack could be easily recognized and added to the blacklist database. As a result, the warning will not only advise the user to stop using at the blacklisted site but also show a safe path to the true site, which is the site that the user intends to go.

Personal information managers try to make the user's online interactions convenient. They securely store the user's passwords and personal information. They automatically fill online forms using the stored data. Users can also copy and paste or drag and drop the stored data from an information manager to any form input. The major security concern of the information manager is how to store the personal information securely. But as far as I know, no information manager bothers to make sure that the current site is good enough to request the user's personal information or is the right site to receive a specific password, as shown in figure 8-12. The information manager already knows the data type that a user is going to submit and if the submitted data is login information, it also knows the user's intended site. It is not hard for the information manager to get the current URL from the browser, especially if it is integrated with a browser. Personal information managers that support online form filling should include a feature to check if the current site meets the user's intention in order to prevent the stored personal data from leaking to a phishing site.

9.2 Future work of Web Wallet

Web Wallet can be improved in terms of both usability and security.

9.2.1 Future work for usability

The usability of Web Wallet can be improved in the following ways.

Protect other types of sensitive information

Web Wallet plans to protect the submission of five types of sensitive information: login information, credit card information, bank account information, personal IDs (e.g., Social Security Number), and personal secrets (e.g., mother's maiden name). These types of information are most often targeted by current phishing attacks. Among them, credit card information, bank account information, personal IDs and personal secrets are types of information that are by themselves sensitive. Their online submission should be better protected than other insensitive submissions. The sites asking for such information should meet some basic criteria, like using SSL to protect the submitted data. Web sites do not meet basic criteria but request such information can reasonably raise warnings because the submission in general is dangerous.

Login information is the sole exception because it is site-specific. Any web site can ask users to register and login as a valid user. The login submission is dangerous only when a user submits the login information to one site but that login account actually belongs to a different site. In order to effectively warn the user about phishing attacks that ask for passwords, the user's intended site should be clear.

Users may reuse the same password at different web sites. It is a bad practice of password usage, and password hashing is proposed to solve this problem. Password hashing can be added as a security feature into Web Wallet, as will be discussed later.

The current Web Wallet protects the submission of both login information and credit card information. Other sensitive types can be protected using the same mechanism used to protect credit card information.

As mentioned in section 6.7, P3P [71] is planned to be integrated into Web Wallet so that Web Wallet can protect all the defined private data types. However, some sensitive data types have not been defined by P3P yet. For example, as for bank account information, P3P specifies the account type, the account number and the account owner's name but not the account bank's name and the account routing number. As a result, an extended P3P specification should be defined and used by Web Wallet. But P3P definition is a good start.

Sensitive input detection

I have shown that Web Wallet can use machine learning mechanisms plus some semantic rules to accurately detect the sensitive inputs in web pages. The big advantage of client-side detection is the Web Wallet protection does not require any change to the web site. It is widely known that any changes required at the web site represent an obstacle to widespread deployment. However, the client-side detection has its own drawbacks. It is not perfectly accurate. As a result, Web Wallet supports user's manual modification of the detection.

One concern is how to train the sensitive detection algorithm with new defined sensitive types. In the current implementation, the login inputs and credit card inputs are manually assigned with sensitive types to form the training set. With every new defined data type, a dedicated team of human experts should train the algorithm and the parameters of the algorithm can then be downloaded by the Web Wallet users. It is better for that team also to monitor the evolution of online submission mechanism by periodically sampling submission forms from the Internet to guarantee the detection algorithm stays accurate.

Web Wallet users can also train the Web Wallet's learning algorithm. When users manually correct the Web Wallet's detection errors, the user's modification can not only be remembered for that particular form but also be used to re-train the detection algorithm of their own Web Wallet. One advantage of this approach is that the learning is customized by the user's personal browsing behavior. Powered by customized learning, the algorithm is likely to be more accurate to that user than the general learning algorithm using the Internet sampling. However, this approach has a vulnerability. Users decide an input's sensitive type mainly based on its visible label, while Web Wallet also uses other hidden parameters (e.g., the field name of the input element) to decide the sensitive type. Attackers can exploit this gap in order to trick users into screwing up their detection algorithm, by letting them mistakenly modify detection errors that are intentionally introduced by the attackers. One way to prevent this attack is to make customized learning based on the site reputation. Only corrections made at the verified sites can be used to re-train the algorithm.

Web site modification can improve the above situation. This is another reason to use P3P, because P3P-enabled sites can easily add the Web Wallet attribute to their sensitive input fields, as shown in figure 6-11. This modification is minor and trivial to implement, compared with Dynamic Security Skins [13] that need the web sites to support the Secure Remote Password protocol (SRP), and Microsoft InfoCard [6] that needs both the web sites and the identity providers to support the new InfoCard submission protocol. XForms, as the next generation of web forms, have already included the P3P data types as a model property (p3ptype) that can be specified for each input. [15] Moreover, with the help of the client-side detection, the web site modification can be done incrementally.

The final goal of Web Wallet is to make sensitive data submission a totally separate channel from other online submissions, so that it can be appropriately protected by the system.

Non-HTML forms

The current Web Wallet can only detect HTML input elements in web pages. Forms displayed by Flash or Java applets are hidden from the browser and thus Web Wallet, because Web Wallet parses the DOM tree provided by the browser. On the other hand, for individual web sites, using non-HTML forms as the only submission mechanism is not common. One reason is accessibility, since screen readers for the blind can understand HTML forms but not Flash forms. Another reason is usability, since it prevents users who do not have the proper plug-ins (like the Flash player) from fully using the site. So, if there is a counterpart HTML form, Web Wallet can use it.

Attackers can also use Flash forms to prevent the sensitive inputs being disabled by Web Wallet. This is another kind of undetected-form attack, which was discussed in chapter 7.

Nomadic users

With the user's permission, Web Wallet can securely store the user's sensitive information on the user's computer. The Web Wallet history is stored locally as well. Currently, users cannot use their Web Wallet data if they switch to a different computer. Sharing the Web Wallet data among multiple computers can be done in two main ways. First is to use a remote service to store and access the Web Wallet data online. Passpet [78] has introduced a remote secure storage mechanism using SRP. The second option is to store the information into a portable hardware device that the user carries around all the time. However, when a user uses a public computer that is not trusted, especially in an Internet Cafe, a trusted personal PDA or a trusted personal cell phone is a better way to use to form a trusted channel with the remote site if the user wants to submit sensitive data to that site. [8, 58]

Site confirmation dialog

Site confirmation proved to be a very useful feature to prevent phishing attacks, but its user interface needs to improve.

The site confirmation dialog lets the user choose his intended site from a site list. To make the confirmation most effective, the site list needs to include the spoofed site under phishing attacks. Currently, the site list is generated from the user's Web Wallet history because many effective phishing attacks spoof the legitimate sites that the user has used before. But it is possible that an attack message can claim to be from an organization with which the user has a relationship in the real world, but has not yet interacted with its online site using Web Wallet. To deal with it, the site list can include the sites that have been recently and commonly spoofed based on some maintained phishing archives, like the one from Anti-Phishing Working Group [30].

On the other hand, the more sites are included in the site list, the less usable the confirmation dialog is, especially the first time a user submits his sensitive data at an unverified but not phishing site. Scrolling over a long list is not desirable. I plan to expand the site display from a one-dimensional list to a two-dimensional area (one dimension being the site reputation and the other dimension being the user's history with the site) so that it can display more sites in a single view.

One security risk that the site confirmation interface should avoid is to make the phishing sites too easy to find and click. In order to improve usability, however, the unverified sites that users intend to go should be easy to find and click. Any new design of the site confirmation interface should be fully studied and tested based on this security and usability tradeoff.

Automatic login

As discussed in section 5.2.3, the stored login cards of Web Wallet also act as bookmarks to the corresponding web sites. A more user-friendly approach is to let users automatically log in to the site using the stored account. The automatic login can be implemented as follows: the first time a user logs in to a site and stores the login information, Web Wallet remembers the login URL, the exact inputs where the login information is filled, and the submit button that the user is clicking. Later when the user triggers the automatic login, the browser is directed to the remembered login URL and runs a script to fill the corresponding inputs and then click the same submit button. This approach fails, however, if the login URL contains a session identifier that can only be used once, or if the site's login form changes. A robust way to do the automatic login is for the web site to provide a secure login web service, so that Web Wallet can call that web service with the account information as parameters.

9.2.2 Future work for security

The security of Web Wallet can be improved in the following ways.

Fake Wallet

The major vulnerability of the current Web Wallet is that its interface can be spoofed – not only the Web Wallet sidebar but also its safe-path suggestion interface. On the other hand, protecting the Web Wallet interface should in theory be an easier problem than protecting arbitrary web sites.

Personal image or customized visual style could be added to personalize Web Wallet to decrease the spoof rate, like [13, 62, 69]. But this is not a fundamental solution because the personal display is essentially like the security indicators that were tested in chapter 3. Checking the personal display is not necessary for the user's current task and may be ignored by the user.

Image recognition techniques could be used to detect the presence of a fake Web Wallet. Image recognition has been proposed to detect phishing attacks by measuring the suspicious pages' visual similarity to the protected pages. [47] However, recognizing web pages has a scalability problem since there are millions of web pages that the user can access. Web sites are also constantly changing their appearance. Moreover, legitimate sites for the same purpose (e.g., bank sites) often intentionally resemble each other to improve their site's usability without actually spoofing or committing fraud. Image detection may mistakenly treat this similarity as faking. Recognizing the Web Wallet interface does not have these problems since it is the only fixed interface that the system needs to recognize. The system knows exactly what Web Wallet looks like and in which part of the screen Web Wallet is located. The Web

Wallet recognition should also be guided by user's feedback. The possibility needs to be minimized that a distorted fake Web Wallet interface is accepted by the user, but fails to be recognized by the system.

Another way to prevent users from typing sensitive information directly into web pages is to constantly monitor the user's keystrokes and compare the string of typed characters with the sensitive information has been submitted through Web Wallet. If part of the string is the same as a password or a credit card number that the user has submitted before, Web Wallet should be automatically opened and the user should be told to switch to Web Wallet.

Two advantages make keyboard monitoring a promising approach. One is that Web Wallet is already intercepting low-level keyboard events in order to zoom out the typed characters. The other advantage is that because users type their sensitive information into Web Wallet at legitimate sites, Web Wallet already knows the sensitive information that it is going to search for. Keyboard monitoring, if implemented correctly, can prevent not only the fake-wallet attack but also the undetected form attack, the two most effective attacks in the Web Wallet security evaluation.

One potential problem of keyboard monitoring is its error rate. For example, a user may use certain common words as his passwords. When he types the same word as part of an insensitive submission, like an email message or a blog message, Web Wallet detects these sensitive words and opens itself by mistake. On the other hand, the backspace key and the delete key could complicate the monitoring because the system does not know which characters and how many characters are removed when these two buttons are typed. As a result, approximate matching should be used instead of perfect matching. But that may in turn introduce more errors that treat insensitive inputs as sensitive.

Negative cues

The high spoof rates of the undetected-form attack and the fake-wallet attack, as shown in chapter 7, indicate that the current implementation of the negative visual cue fails. Moreover, a common objection to the character zooming is that it will increase the risk of shoulder-surfing.

One possible problem is that the negative cue is introduced too commonly: it appears whenever a user types into a web page. The purpose of the negative cue is to indicate an unsafe mode to the user. Since Web Wallet fully trusts verified sites, the negative cue could be disabled when users are interacting with the verified sites. With less frequent appearance, the negative cue is expected to be more effective and less annoying. Moreover, other types of visual feedback other than zooming should be designed and tested.

Aside from being a security feature of Web Wallet, negative cues can be applied independently to differentiate a safe mode from a unsafe mode, for example, SSL-protected web pages from non-SSL ones. When a user types into a non-SSL web page, the typed characters could be reflected. Compared to the current SSL indicators in the peripheral area of a browser, this negative cue is always in the center of the user's field of vision.

Attack targets the secure storage

Because Web Wallet stores the user's sensitive data, its storage could be an attractive target to attackers. A virus or worm can search the user's computer for the stored data. The current Web Wallet uses the standard cryptographic services provided by Microsoft .NET framework. The stored data is encrypted using an AES key generated from the user's master password. Passpet introduces a smart way with a usable interface that helps users to generate a strong master secret from their master password and use that secret to create user's site-specific passwords. Web Wallet can use the same mechanism to make the encryption key stronger in order to prevent the brute force attacks. Secure file systems, like the coming Windows Vista BitLocker Drive Encryption, can also be used to securely store the Web Wallet data.

Integrating with password hashing

Password hashing is a useful feature to prevent users from using weak passwords and sharing the same password among multiple web sites. Since Web Wallet is used when a user registers an account and updates an existing account, Web Wallet could provide an option so that the user can choose to let Web Wallet generate different site passwords from his master password using the password hashing mechanism.

Password hashing can make users fully depend on Web Wallet to submit sensitive information because they do not even know their passwords for different web sites. As a result, they have to use the saved login card from the authentic Web Wallet. Both fake-wallet attacks and undetected-form attacks can be effectively prevented.

Pharming

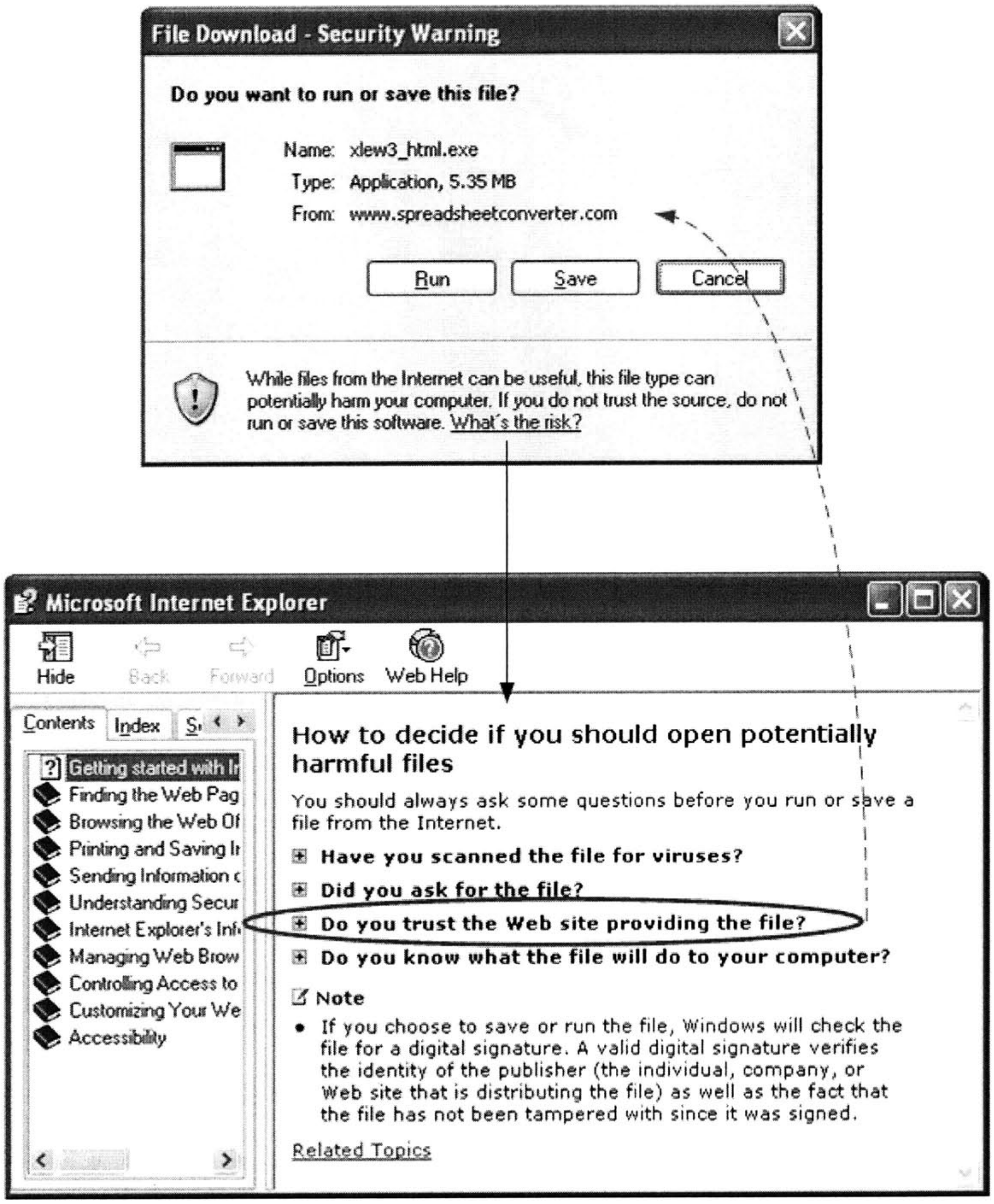
Pharming is an improved version of phishing. The pharming site not only duplicates the look and feel of the targeted legitimate site, but also displays the same hostname as the legitimate one in the browser's address bar. Pharming is typically done through poisoning the victim's DNS server, remapping the target website's domain to the IP address of the pharming site. In this case, even if the victim enters the web address properly into his browser, he will still be directed to the pharming site.

Domain Name System Security (DNSSEC), which requests digital signatures for all Internet communications, is the fundamental solution to pharming, but it needs a major change to the Internet as a whole. Active cookies [40] are another solution by preventing an attacker from impersonating a user on the legitimate site because the attacker does not have the cookie that is installed into the user's browser when he first visits the site and is then used to identify the user's browser to the site. But it does not protect the user from being tricked and providing his information to the attacker.

Web Wallet can be implemented to prevent pharming attacks. Whenever Web Wallet is open, which means that the user wants to submit sensitive data, Web Wallet can check if the current IP address matches the displayed hostname in the address bar using a trusted reverse DNS lookup service secured and authenticated by SSL. The IP-checking can be integrated with the site reputation request process mentioned in section. The IP-checking is expected to be an expensive process and is only started when the user submits sensitive data through Web Wallet, which should be given some extra protection.

Malware-based phishing

Instead of tricking users into submitting their sensitive information online, malware-based phishing asks users to download and install viruses that are claimed to be useful software like security patches. The design principles of Web Wallet can be used to prevent malware-based phishing. Figure 9-1 shows the security warning of Internet Explorer (IE) for file downloading and the help document that explains the risks of file downloading. The help document says that users should ask four question before run or save a file from the Internet. One interesting question is "do you trust the web site providing the file". The IE warning shows the site URL. It acts as a security visual indicator that has been tested to be ineffective. Even worse, because only the hostname of the downloading site is displayed, the attacks that use similar hostnames can easily trick users. The site confirmation dialog of Web Wallet can be used here to ask users explicitly indicate the downloading site. The site list will include the current site, a set of sites has been recently and commonly spoofed by malware-based phishing, and the sites where users have downloaded software before. Usability can be improved to let users only confirm the downloading site using the site confirmation dialog when the site is unverified and the downloading file is not correctly signed.



File Download - Security Warning

Do you want to run or save this file?

Name: xlew3_html.exe
Type: Application, 5.35 MB
From: www.spreadsheetconverter.com

While files from the Internet can be useful, this file type can potentially harm your computer. If you do not trust the source, do not run or save this software. What's the risk?

Microsoft Internet Explorer

How to decide if you should open potentially harmful files

You should always ask some questions before you run or save a file from the Internet.

Have you scanned the file for viruses?

Did you ask for the file?

Do you trust the Web site providing the file?

Do you know what the file will do to your computer?

Note

- If you choose to save or run the file, Windows will check the file for a digital signature. A valid digital signature verifies the identity of the publisher (the individual, company, or Web site that is distributing the file) as well as the fact that the file has not been tampered with since it was signed.

Related Topics

Figure 9-1: Internet Explorer security warning for file downloading

Bibliography

[1] B. Adida, S. Hohenberger, and R. Rivest. Lightweight Encryption for Email. In *USENIX Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI)*, 2005.

[2] D. Bank. 'Spear Phishing' Tests Educate People About Online Scams. The Wall Street Journal, August 2005.

[3] P. Behera and N. Agarwal. A confidence model for web browsing. In *Toward a More Secure Web - W3C Workshop on Transparency and Usability of Web Authentication*, 2006.

[4] Fight Identity Theft Blog. ING Direct Fights Keystroke Loggers. http://fightidentitytheft.com/blog/?p=23, December 2005.

[5] M. Bolin. End-user Programming for the Web. Master's thesis, Massachusetts Institute of Technology, June 2005.

[6] K. Cameron and M. Johns. Design Rationale behind the Identity Metasystem Architecture. http://www.identityblog.com/wp-content/resources/design_rationale.pdf, 2006.

[7] N. Chou, R. Ledesma, Y. Teraguchi, and J.C. Mitchell. Client-Side Defense Against Web-Based Identity Theft. In *11th Annual Network and Distributed System Security Symposium*, 2004.

[8] D. Clarke, B. Gassend, T. Kotwal, Burnside M., M. Dijk, S. Devadas, and R. Rivest. The Untrusted Computer Problem and Camera-Based Authentica-

tion. In *Proceedings of the International Conference on Pervasive Computing*, 2002.

[9] J. Coco, S. Klein, D. Schutzer, S.Y. Yen, and A. Slater. Using P3P for E-Commerce. http://www.w3.org/TR/P3P-for-ecommerce, November 1999.

[10] CoreStreet. SpoofStick. http://www.spoofstick.com/, 2004.

[11] G. Cybenko, A. Giani, C. Heckman, and P. Thompson. Cognitive Hacking: Technological and Legal Issues. In *Law and Technology*, 2002.

[12] M. Delany. Domain-based Email Authentication Using Public-Keys Advertised in the DNS (DomainKeys). Internet Draft, September 2005.

[13] R. Dhamija and J.D. Tygar. The Battle Against Phishing: Dynamic Security Skins. In *Symposium On Usable Privacy and Security*, 2005.

[14] R. Dhamija, J.D. Tygar, and M. Hearst. Why Phishing Works. In *Conference on Human Factors in Computing Systems*, 2006.

[15] M. Dubinko. *XForms Essentials.* O'Reilly, 2003.

[16] EarthLink. Earthlink Toolbar. http://www.earthlink.net/software/free/toolbar/.

[17] D. Eastlake. Electronic Commerce Modeling Language (ECML) Version 2 Specification. RFC 4112, June 2005.

[18] eBay. Tutorial: Spoof (fake) Emails. http://pages.ebay.com/education/spooftutorial/.

[19] eBay. Using eBay toolbar's account guard. http://pages.ebay.com/help/confidence/account-guard.html.

[20] A. Emigh. Online Identity Theft: Phishing Technology, Chokepoints and Countermeasures. In *ITTC Report on Online Identity Theft Technology and Countermeasures.*, October 2005.

[21] F-SECURE. Hoax warnings: Sulfnbk.exe virus hoax. http://www.f-secure.com/hoaxes/sulfnbk.shtml, May 2001.

[22] FDIC. Putting an end to account-hijacking identity theft. http://www.fdic.gov/consumers/consumer/idtheftstudy/identity_theft.pdf, 2004.

[23] Department of Justice Federal Bureau of Investigation. FBI Says Web Spoofing Scams Are a Growing Problem. http://www.fbi.gov/pressrel/pressrel03/spoofing072103.htm, 2003.

[24] E.W. Felten, D. Balfanz, D. Dean, and D.S Wallach. Web Spoofing: An Internet Con Game. In *20th National Information Systems Security Conference*, 1996.

[25] S. Fluendy. Phishing targeting online outlets. Computer Crime Research Center. http://www.crimeresearch.org/news/03.16.2005/1050/, March 2005.

[26] G.D. Forney. The Viterbi algorithm. In *IEEE*, March 1973.

[27] S. Garfinkel and R. Miller. Johnny 2: A User Test of Key Continuity Management with S/MIME and Outlook Express. In *Symposium On Usable Privacy and Security*, 2005.

[28] GeoTrust. TrustWatch Search – Tools, Toolbars, Extensions and more... http://www.trustwatch.com/, 2004.

[29] Google. Google Safe Browsing for Firefox. http://www.google.com/tools/firefox/safebrowsing/index.html, 2005.

[30] Anti-Phishing Working Group. Phishing Archive. http://www.antiphishing.org/phishing_archive.html.

[31] Anti-Phishing Working Group. eBay - NOTICE eBay obligatory verifying - Invalid user information. http://www.antiphishing.org/phishing_archive/eBay_03-09-04.htm, March 2004.

[32] Anti-Phishing Working Group. Phishing activity trends report, June 2006. http://antiphishing.org/reports/apwg_report_june_2006.pdf, June 2006.

[33] A. Herzberg. The Unprotected Login Inter-Net Fraud League (I-NFL) Hall of Shame. http://www.cs.biu.ac.il/~herzbea//shame/, 2005.

[34] A. Herzberg and A. Gbara. TrustBar: Protecting (even nave) web users from spoofing and phishing attacks. http://www.cs.biu.ac.il/ herzbea/Papers/ecommerce/spoofing.htm, 2004.

[35] D. Huynh, R.C. Miller, and D.R. Karger. Breaking the Window Hierarchy to Visualize UI Interconnections. http://people.csail.mit.edu/dfhuynh/research/research.html, 2004.

[36] iS3. ZILLAbar Anti-Phishing Toolbar. http://www.zillabar.com/, 2006.

[37] T. Jagatic, N. Johnson, M. Jakobsson, and F. Menczer. Social Phishing. In *Communications of the ACM*, 2006.

[38] A. Jesdanun. Keystroke Logger Captures Passwords At Copy Shop: Experts Advise Caution. Associated Press, 2003.

[39] M. Johns. A guide to supporting infocard v1.0 within web applications and browsers. http://www.identityblog.com/?page_id=412#infocardg_topic5a, March 2006.

[40] A. Juels, M. Jakobsson, and T. Jagatic. Cache Cookies for Browser Authentication (Extended Abstract). In *IEEE Security and Privacy*, 2006.

[41] B. Krebs. Worm comes disguised as windows warning - new Internet worm disguises itself as official virus warning from Microsoft Corp. WashingtonPost, September 2003.

[42] B. Krebs. The New Face of Phishing. The Washington Post, February 2006.

[43] K. Krebsbach. Goin' Phishing: Growing e-mail attacks threaten banks' bottom lines, April 2004.

[44] Stanford Persuasive Technology Lab. Stanford Web Credibility Research. http://credibility.stanford.edu/.

[45] J. Leyden. Crooks harvest bank details from Net kiosk. The Register. http://www.theregister.co.uk/2003/01/27/crooks_harvest_bank_details/, 2003.

[46] A. Liten. Increased Phishing and Online Attacks Cause Dip in Consumer Confidence, June 2005.

[47] W. Liu, X. Deng, G. Huang, and A.Y. Fu. An Antiphishing Strategy Based on Visual Similarity Assessment. In *IEEE Internet Computing, Vol. 10, No. 2,* "March/April" 2006.

[48] A. McCallum and K. Nigam. A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI*, 1998.

[49] D. McCullagn. Ex-student accused of spying on campus. CNET, 2003.

[50] D. McFarland and A. Mongrain. Atlanta, GA; Detroit, MI; and Austin, TX ring the loudest when it comes to cell phone ownership. In *Scarborough Research*, 2003.

[51] Microsoft. What is spear phishing? Help prevent identity theft from new, targeted phishing scams. http://www.microsoft.com/athome/security/email/spear_phishing.mspx, December 2005.

[52] Netcraft. Netcraft Toolbar. http://toolbar.netcraft.com/, 2004.

[53] D.A. Norman. *The design of everyday thing.* Basic Books, 2002.

[54] Passfaces. A Phishing Story. http://www.realuser.com/published/A%20Phish %20Story.pdf, 2005.

[55] J. Rennie, L. Shih, J. Teevan, and D.R. Karger. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. In *Twentieth International Conference on Machine Learning*, 2003.

[56] RoboForm. Roboform features: Fill in forms, automatic forms completion, save forms, search toolbar. http://www.roboform.com/features.html.

[57] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J.C. Mitchell. Stronger Password Authentication Using Browser Extensions. In *Proceedings of the 14th Usenix Security Symposium*, 2005.

[58] S.J. Ross, J.L. Hill, M.Y. Chen, A.D. Joseph, D.E. Culler, and E.A. Brewer. A Composable Framework for Secure Multi-Model Access to Internet Service from Post-PC Devices. In *Third IEEE Workshop on Mobile Computing Systems and Applications*, 2000.

[59] B. Schneier. Semantic Attacks: The Third Wave of Network Attacks. In *Crypto-Gram Newsletter*, October 2000.

[60] W3 Schools. Browser Statistics. http://www.w3schools.com/browsers/browsers_stats.asp, 2006.

[61] Fujitsu SDA. mPollux SMS Security Option, 2003.

[62] PassMark Security. Two-Factor Two-Way Authentication. http://www.passmarksecurity.com/.

[63] RSA Security. RSA Mobile: two factor authentication for a mobile world, 2002.

[64] T. Sharif. Phishing Filter in IE7. http://blogs.msdn.com/ie/archive/2005/09/09/463204.aspx, September 2006.

[65] Tropical Software. Secure Browser for Windows. http://www.tropsoft.com/secbrowser/.

[66] C. Spiezle. Authentication Identity Crisis. In *Email authentication implementation summit 2005*, July 2005.

[67] B. Sullivan. Consumers still falling for phish. http://www.msnbc.msn.com/id/5519990/, July 2004.

[68] A. Treisman and S. Gormican. Feature analysis in early vision: Evidence from search asymmetries. In *Psychological Review, 95*, 1988.

[69] J.D. Tygar and A. Whitten. WWW Electronic Commerce and Java Trojan Horses. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, 1996.

[70] Ubisecure. Ubilogin The Extensive Cross-Platform Single Sign-On Solution, 2003.

[71] W3C. Platform for Privacy Preferences (P3P) Project. http://www.w3.org/P3P/.

[72] T. Whalen and K. Inkpen. Gathering Evidence: Use of Visual Security Cues in Web Browsing. In *Graphics Interface*, 2005.

[73] A. Whitten and J.D. Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *8th Usenix Security Symposium*, 1999.

[74] Z. Ye and S. Smith. Trusted Paths for Browsers. In *ACM Transactions in Information Systems Security 8:2*, May 2005.

[75] Z. Ye, Y. Yuan, and S. Smith. Web Spoofing Revisited: SSL and Beyond. In *Technical Report TR2002-417, Dartmouth College*, 2002.

[76] K.P. Yee. Challenges: Simon Says. http://usablesecurity.com/2005/07/23/simon-says/, July 2005.

[77] K.P. Yee. Guidelines and Strategies for Secure Interaction Design. In *Security and Usability: Designing Secure Systems that People Can Use.* O'Reilly, 2005.

[78] K.P. Yee and K. Sitaker. Passpet: Convenient password management and phishing protection. In *Symposium On Usable Privacy and Security*, 2006.

[79] T. Zhang and F.J. Oles. Text categorization based on regularized linear classification methods. In *Information Retrieval*, 2001.