iPhish: Phishing Vulnerabilities on Consumer Electronics

Yuan Niu, Francis Hsu, Hao Chen
University of California, Davis

Abstract

As consumer electronic devices with embedded browsers become popular, financial institutions and online merchants set up websites to accommodate visitors using these devices. These devices range from cell phones to gaming consoles, cars, and even refrigerators. Porting a traditional desktop[1] browser to a mobile device is more involved than resizing the display. To adapt to the hardware limitations inherent in mobile devices, mobile browsers often remove or replace certain features commonly found in traditional browsers. Unfortunately, some of these features are critical for depending against phishing attacks. We studied browsers on three mobile devices and discovered vulnerabilities in their input, chrome, and URL display. We conducted a user study to confirm our findings on the iPhone Safari browser, one of the most popular mobile browser platforms. For each potential vulnerability, we were able to construct a phishing scenario to successfully fool users into giving away the credentials for a role-played Bank of America account. To mitigate the vulnerabilities, we propose to designate and display URLs in a more phishing-resistant way, and to create an anti-phishing proxy that is independent of mobile devices or browsers.

1 Introduction

Web browsers started as applications running on desktop and laptop computers. Users sat in front of a relatively large screen with a keyboard and mouse. Today, the web is evolving into a general computing platform where the user can access most of his computing resources. Due to the popularity and ubiquity of the web, web browsers reach beyond traditional desktops and laptops to enter many non-traditional computing platforms, such as mobile phones (e.g., iPhone), video game systems (e.g., Nintendo DS and Nintendo Wii), and even refrigerators [2].

As web-capable devices make their way into consumers' hands, services, such as banking and shopping, are being adapted for these new platforms. For example, Bank of America [2] and Amazon [1] have introduced websites for mobile devices, indicating the emerging paradigms of e-commerce on less traditional computing devices.

However, adapting websites and web browsers for consumer electronic devices may introduce unexpected security vulnerabilities. Due to hardware limitations on these devices, browsers often have to remove or replace features. If users rely on these features for detecting phishing websites, the users will be more vulnerable to phishing on these browsers. For example, due to typically small display sizes on these devices, their browsers may hide, or allow the web page to hide, the URL bar or the status bar. Even when they display URLs, they truncate long URLs to fit within the screen width. Since the ability to read and vet URLs is crucial for defending against phishing attacks, without this ability ordinary users who have trouble parsing URLs are still just as prone to phishing, and even expert users who can parse URLs will find it more difficult to detect phishing websites. Moreover, certain convenient features on desktop or laptop computers encourage users to use web browser in a way that is less vulnerable to phishing. When consumer devices remove these convenient features, often due to physical limitations, users tend to behave in ways that are more vulnerable to phishing. For example, many consumer electronic devices replace physical keyboards with soft keyboards on the screen. Since typing on these soft keyboards is often unfamiliar, slow, and inaccurate, users are discouraged to type URLs manually, which is less vulnerable to phishing, and are encouraged to follow links, which are more vulnerable to phishing.

We examined three consumer electronics devices:

[1] For the sake of succinctness, we refer to browsers run on desktop and laptop machines as simply "traditional browsers."

[2] http://us.lge.com/products/model/detail/home\%20appliances_refrigerators_side-by-side_LSC27991.jhtml

iPhone, Nintendo DS, and Nintendo Wii. We identified their hardware limitations, their browser limitations, and the impact of the limitations on phishing attacks (Section 2). Then, we set up a user study to evaluate the vulnerability of users to phishing attacks on iPhone (Section 3). We describe the findings from our user study in Section 4. Finally, we propose approaches for both browsers and mobile websites to mitigate phishing attacks (Section 5).

2 Devices

We examined three consumer electronics products: iPhone, Nintendo DS and Nintendo Wii. The iPhone runs a modified version of the Apple Safari browser, while the Nintendo DS and Wii use modified Opera browsers. The modifications to the browsers accommodate the limitations inherent to the new platforms, as described below.

2.1 Display

The modified browsers run on platforms with displays that are much smaller than those used for desktop browsers. While typical desktop browsers run on displays of 1024x768 pixels or larger [10], the displays of the devices that we examined are smaller due to portability or compatibility constraints. Table 1 lists the display resolutions of the platforms. Note that the total display area is at between 12.5%-40% of a desktop browser.

Platform	Display Resolution
Apple iPhone Safari	320x480
Nintendo Wii Opera	640x480
Nintendo DS Opera	256x384

Table 1: Maximum display resolution of devices

A browser window normally consists of control menus, status displays, a current URL display, and content pane of the current site. Since a user primarily interacts with the content pane, it logically follows that a browser would maximize the content pane at the expense of other elements on the browser window, in response to the display limitation. For example, traditional browser chrome elements (the portions of a browser window that do not display Web page content) normally hold the entire URL of the displayed site unless the URL is unusually long. The Nintendo DS reserves a small fixed 256 x 15 pixel bar at the top of the display for the URL, displaying about 22 characters across in a small font. The iPhone uses a 320 x 60 pixel bar for the URL, but the URL text only occupies an area 240 pixels across. Moreover, iPhone's URL bar can be scrolled off the browser windows, either by the user or by a script in the webpage, as if the URL bar is part of the webpage content. The Nintendo Wii automatically hides URL bar completely when displaying the page content.

2.2 Input

Without the traditional input devices of keyboard and mouse, these platforms require the user to navigate by pointing. The iPhone and Nintendo DS both use a touch screen interface while the Nintendo Wii uses a wand controller to point at the screen controls. They all use an on-screen keyboard that allows users to peck out input text characters one by one. Compared to using a keyboard, inputting text in this manner is stranger, slower, and less precise.

2.3 Software Updates

All three browsers currently run on restricted software platforms, which limit the applications that can run there. While this restriction may be motivated by security since it may prevent malware from attacking the system, it also prevents the user from adding useful software, such add-ons that customize or extend the browser. Furthermore, users cannot apply security updates to these browsers as easily as to desktop browsers. The iPhone requires its user to dock it with a computer to apply software updates [5]. The Nintendo Wii requires navigating through a lengthy setup menu. The Nintendo DS distributes its browser software only on a read-only memory cartridge, thus preventing updates from the Internet completely.

3 User Study Setup

We conducted a study to compare users' reactions to similar phishing scenarios on a desktop Safari browser and the iPhone Safari browser. We recruited three groups of 37 participants total:

1. *Unscreened users*. We recruited 11 unscreened users, from Facebook, Craigslist, and flyers.

2. *Knowledgeable users* We recruited 6 knowledgeable users from undergraduate computer science classes who had heard the term *phishing* and who knew what indicators to look for in identifying phishing attacks. Four of our participants owned an iPhone or iPod touch. Out of the four, two were able to detect phishing attacks, and the other two failed.

3. *Expert users* We recruited 20 expert users from a graduate-level computer security class. As an indication to their security sophistication, all but three identified all the phishing attacks.

To protect the privacy of our participants, we created a fake persona and cloned the mobile and desktop versions

of the Bank of America website. We modified the DNS setting to route all requests for *bankofamerica.com* to our cloned sites. We changed settings in the desktop Safari browser to suppress certificate warnings but were unable to do so on the iPhone. We instructed the participants to play the role of a user, and to access the user's Bank of America and Facebook accounts using the credentials that we provided.

Before the user started the tasks, we gave them a brief tutorial of the iPhone's features. Specifically, we highlighted the thumbnail view's display of the page title and URL, and how to switch between landscape and portrait modes.

We asked users to perform a series of tasks involving a Bank of America account on both the iPhone Safari browser and the desktop Safari 3 beta browser. The first tasks on both browsers was to log into the "real" Bank of America websites and record an account balance. This was intended to be a control task to familiarize users with the process of logging in and to show them what the real Bank of America website looks like. After performing the control task, we asked the users to check a Gmail account for further instructions via e-mail. The user first did the tasks on a laptop using the Safari 3 browser and then continued the tasks on iPhone Safari browser. We maintained the order so that users could have a chance at recognizing the more obvious phishing tactics in a familiar environment before moving on to similar tasks on a new device. We hoped to prime the users by alerting them of possibly suspicious behaviors.

On the Safari 3 browser on the laptop, we instructed the user, via emails, to visit the following phishing websites:

1. A website that hides the browser's URL bar.
2. A website with an incorrect domain.

On the iPhone, after visiting the above websites, the user was instructed to visit the following phishing websites to exploit iPhone specific weaknesses:

1. A website that displays a fake browser chrome.
2. A website with an incorrect domain and a long URL such that the displayed URL in iPhone fails to show the incorrect domain (Section 4.2.1).

We did not create desktop versions of these iPhone websites because they exploit problems we think are iPhone specific. Duplicating the chrome of a desktop browser on a website convincingly is considerably harder for phishers than duplicating the chrome of the iPhone browser. As for long URLs, a typical desktop browser has at least 800 pixels for text in the URL bar, and users can use a mouse to easily scroll through the URL. Given the relatively small width (less than 400 pixels) of the URL bar on the iPhone and the lack of any input devices besides fingers, users cannot see the entire URLs. As a further barrier, scrolling through the URL manually is available only in URL input mode on iPhone.

The users' instructions included an admonition intended to alert to the fact that they should be concerned with security: *It is **NOT NECESSARY** to complete each task. Only complete the task if it DOES NOT compromise the security of Neo's account information.*

4 Vulnerabilities

The need to conserve screen real estate on the iPhone led to the sacrifice of browser chrome features in Safari. There is no bottom status bar because the iPhone does not support `mouseover()` events. If a user holds down the link for a few seconds, they see the destination URL in a hover text. The static bottom bar contains navigation buttons, bookmarks, and thumbnail views. We examine some specific points of weakness below and present scenarios in which these weaknesses could be exploited.

4.1 User Input

Typing is a tedious and often inaccurate process for users not accustomed to such a tiny touch screen touchpad. Because of this, it is tempting to follow links in e-mails rather than typing the links manually. Every user new to the iPhone in our study expressed frustration with typing, especially in password fields.

Additionally, the iPhone provides no shortcuts to quickly switch from one application to another. To switch from Mail to Safari, for example, the user must click to go back to the main screen, which needs time to render, and switch to a new application, which must load, taking at least two clicks. Switching from a mail application to a browser on a computer or laptop, however, requires no more than one click or a simple Alt-Tab or Apple-Tab almost instantaneously. Even switching between browser windows on the iPhone is more complicated. iPhone users must first initiate thumbnail mode and choose the correct window, clicking at least once and scrolling through at least one other window, and tapping on that window. A click of the mouse or another Alt-Tab/Apple-Tab, again, performs this simple task on the desktop browser almost instantaneously. On the iPhone, switching windows from one application to another, or even within one application, is a much longer and more complicated process than its equivalent on desktops. Users who value convenience have great temptation to follow links from other applications.

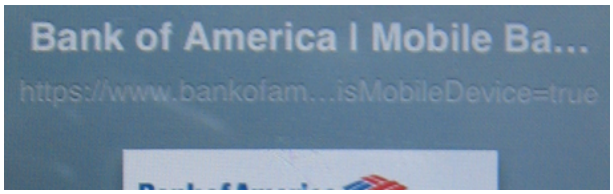
Five average users said they would not usually follow links in emails on desktop computers, but that typing in the iPhone browser was "annoying". Moreover, the iPhone made it more tempting for them to click the links provided in emails when they faced with the tedious alternative of reading an e-mail, observing the link, nav-

igating to the home screen to reopen Safari, and finally typing the URL.

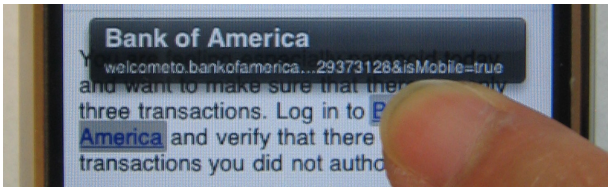
4.2 URL Display

4.2.1 URL Truncation

A fundamental flaw in the iPhone Safari browser lies in its URL display. All URLs too long to fit on one line are truncated. In thumbnail view (Figure 1(a)), a substantial middle portion of the URL is truncated and replaced with an ellipsis with no way to expand the truncated portion of the URL. The only way by which a user can view the entire URL is to go back to the URL input and manually scroll, letter by letter, through the URL. In hover view, the middle portion of the URL is truncated in the same way as in the thumbnail view (Figure 1(b)). Furthermore, the onhover destination link is displayed after a few seconds of holding down the link. Upon release, the hover action may still be interpreted as a click if the user lifts rather than slides the finger away from the link, causing the browser to load the phishing page.



(a) Thumbnail Mode



(b) Link Hover

Figure 1: Display of truncated URLs

An attacker from a rogue domain wishing to impersonate a legitimate domain may simply construct a long URL that uses the legitimate domain name as a subdomain name. For example, a phisher wishing to deceive Bank of America users could construct a website on `phishydomain.com` by creating a subdomain ending with `bankofamerica.com` and using long filenames as in the following URL: `subdomain.bankofamerica.com.phishydomain.com/longfilename`. The attacker can select the string `subdomain` in the above URL such that the beginning part of the URL — `subdomain.bankofamerica.com`, which appears legitimate — is displayed in the browser's URL bar but the next string — `.phishydomain.com`, which is untrusted — is truncated in the URL bar. The URL bar uses about 50 pixels to display the `http://` or `https://` portion of the URL. Assuming a conservative estimate of 10 pixels per letter, to hide `phishydomain.com` in portrait mode, we need a subdomain name of about seven letters to prepend to `bankofamerica.com.phishydomain.com` because `bankofamerica.com` uses about 170 pixels. In instances where the SSL lock icon appears, subdomain names can be even shorter.

4.2.2 Address Bar

As a feature to maximize the screen space for content, the iPhone Safari browser scrolls the address bar along with the page content. Web page creators can also use a Javascript `scrollTo()` trick to hide the URL bar on page load by jumping to a predetermined location within the page. According to the iPhone user manual [5], tapping on the status bar at the top of the screen brings the user back to the top of the page, but the same `scrollTo()` trick can be used with DOM information to always jump to a predetermined location once the user scrolls too close to the address bar. This, in effect, nullifies the built in function to show users what URL they're at. Users can still view the URL in thumbnail mode, however.

The iPhoneWebDev Google group [3] contains guides and threads discussing various way to hide the URL bar. One developer, listed "Hiding the URL bar ... more permanently" as the third priority point [4] to bring up with Apple iPhone developers at a tech talk.

Hiding the URL bar on page load is widely used and accepted, as evidenced by the Bank of America [5], Amazon, and Facebook mobile versions. A phisher can easily take advantage of this by hiding the URL of the page on page load. To fool users who are more vigilant and observant during page load, the phisher can use a very long URL or a URL with a slight difference, such as "vv" in place of "w" or "1" in place of "l." In our user study task that hid the URL bar, none of the average users or knowledgeable users was alerted by the missing URL bar and proceeded without verifying the URL. When asked about this during debriefing, one user responded, "It hampers the usability to always check." The same user noted that because he could not see the URL, it "did not allow the room to be suspicious."

[3] http://groups.google.com/group/iphonewebdev

[4] `http://groups.google.com/group/iphonewebdev/browse_frm/thread/e98bcb426a036d3a`

[5] The mobile version of Bank of America we spoofed was a general mobile version, and not the new iPhone version.

4.2.3 Weaknesses in the Opera browser

In our examination of the Opera browser on the Nintendo DS and Wii, we noticed problems in the URL display. Although banking from a gaming console is currently rare, it may become popular with the release of a keyboard. As an indication, financial transactions are already taking place through gaming consoles on the XBOX Live network and Wii marketplace.

As shown in Figure 2, the Wii Opera browser simply does not display the URL with the current page at all. The is no way to preview the destination of a URL link on a page. To view the current URL of a page, the user must take an extra action and click on the page information button on the chrome toolbar. The information page displays the page address on one line. The user must manually scroll through a URL that cannot fit in one line of the display width. Information about encryption being used by the page is left out of the URL since it lacks the `http/https` portion.

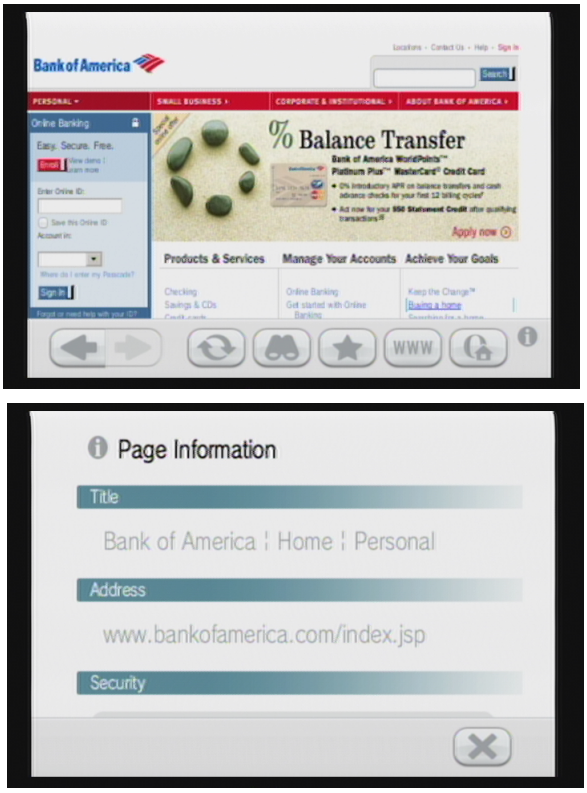


Figure 2: The Opera Wii Browser display and information page

4.3 Chrome Simplicity

Compared to a traditional browser, the iPhone Safari browser's chrome is relatively easy to game. Users of traditional browsers can choose the software they use and further customize that software with add-ons and bookmarks. Users of the iPhone browser can only add bookmarks, which are displayed from a button on the immutable bottom menu.

An attacker can easily compromise the use of the address bar as a location indicator by a simple Javascript `scrollTo()` trick described in Section 4.2.2. The trick can force the page to jump back to the location of the fake URL bar on the webpage. However, this trick also causes the page to behave strangely when a user attempts to scroll manually, because the page always jumps back to the top automatically, and the user can defeat this trick by checking the page's URL in the thumbnail view. When users encountered the strange scrolling behavior, however, they consistently tried to complete the task and made no comment except to express annoyance. One, for example, wrote "Hate interface" as feedback. Another user was unusually tenacious and typed in the password despite having trouble seeing the form field as a result of the odd scrolling behavior.

Users from all groups, even expert users, who identified the URL as phishing by looking at the URL, failed to notice the fake address bar sliding over the real one very quickly on page load. Average and knowledgeable users who tried to proceed in this stage should have noticed it multiple times, but none did until it was specifically pointed out in the debriefing stage.

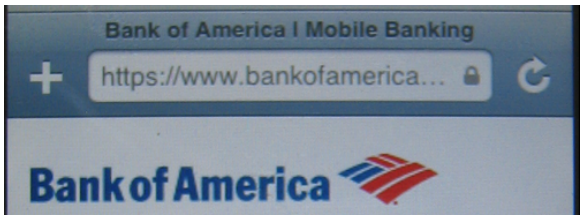
Creating a false address bar is also quite easy: simply using HTML and Javascript within the page, a phisher can replicate all the features of the address bar except for the "Add Bookmark" button. While no users actually tested this feature, we believe that omitting this feature will cause users to believe that it is an iPhone glitch based on their feedback in encountering scrolling problems. Figure 3 shows the real iPhone chrome and our fake chrome.

In our user study, even expert users admitted how convincing the fake chrome looked. One expert user admitted that had he not been primed by prior instances of phishing, he would have fallen for the fake chrome trick. Another expert user simply wrote "I was fooled." Only one user, out of all 37, examined the chrome closely enough to realize that it was fake. He commented that the SSL lock icon in the address bar did not look quite right.

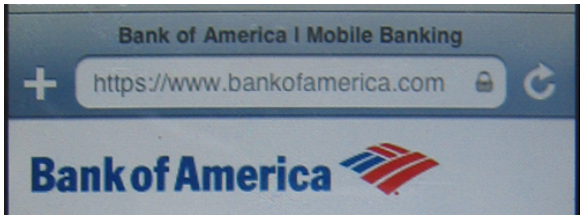
4.4 SSL

When a user navigates to a site that does not have a correct SSL certificate, the Safari browser pops up a short dialog with options to either continue or abort. There is no option for the user to examine the certificate and no explanations provided as to why the certificate is invalid.

Although the URL bar displays a lock icon when the page loaded uses SSL, average and knowledgeable users still tended to look for a lock icon within the rendered HTML page.



(a) Real Chrome



(b) Fake Chrome

Figure 3: Side by side comparison of the real iPhone chrome with our fake chrome.

Expert users refused to proceed on tasks after noting the lack of SSL. They also refused to proceed when they encountered an invalid certificate error. They complained repeatedly that because there was no way for them to check the details of a certificate, they were not comfortable proceeding. Average and knowledgeable users, on the other hand, ignored the invalid certificate error. Only two average users did not proceed after seeing the SSL error.

The SSL error affected just the first task directly, since subsequent tasks were designed to test the effectiveness of our spoofing. It served to penalize us because it should have acted as a primer to warn users of possible phishing attempts. Nevertheless, it was not enough to deter most non-expert users.

4.5 Mail Application

We used Gmail for all our emails. We discovered that while the browser based version of Gmail performed phishing filtering and injected a warning in red, the iPhone Mail application did not. If users do not examine email headers, the version of the phishing email in the iPhone Mail application contains a seemingly valid sender address without any warnings. The iPhone Mail application also does not strip suspicious link tags, while the browser based version of Gmail does. This, in combination with a long URL, can spoof a real online banking e-mail notice convincingly. While this problem is not specific to the iPhone, it is harder to mitigate. Desktop mail clients can use plugins and user defined filters, but the iPhone Mail application is immutable except when Apple releases firmware upgrades.

5 Mitigation

Since browser users on consumer electronic devices are more susceptible to phishing attacks, browser developers, website designers and users should take steps to mitigate the attacks. In this section we propose these approaches.

5.1 URL Designation and Display

To make phishing more difficult, website designers can create more recognizable sites, and web browsers can make the origin and authenticity of the site more apparent to the user.

One focus point for website and browser designers is the URL since it reliably identifies the domain of a page. Website designers can help by shortening URLs so that they appear completely on short URL bars on consumer electronic devices. This could also help users parse and evaluate the authenticity of the URL. The iPhone Safari browser could be improved if more screen real estate is sacrificed so that the address bar is a permanent part of the chrome. Prohibiting webpages from tampering with the URL bar would prevent them from hiding the bar and displaying a fake URL in the page. When a long URL exceeds the width of the URL bar, the iPhone Safari browser should scroll the URL, as what the desktop browser does, instead of truncating the URL to prevent the attacker from hiding their malicious domain name in the URL. If truncation is unavoidable, the URL bar should display the most important part of the URL, typically the second level domain (e.g. `bankofamerica` in `www.bankofamerica.com`), that helps establish the real identity of the website.

5.2 Proxies

One way to shrink the gap between desktop and mobile browsers is to bring plugins and more processing power to the consumer mobile devices. Since it is difficult to upgrade the hardware of these devices, we accomplish the above goal through a proxy service that is independent of mobile devices' platforms and browsers. A major limitation of the iPhone and other mobile browsers is the difficulty or inability for users to download updates and plugins. We have designed an Internet Content Adaptation Protocol (ICAP) [4] server to be used with a standard web proxy to protect users from phishing sites. Our proxy service requires a one-time configuration in the browser to pass requests through the proxy server instead of directly retrieving pages. The proxy server then passes the request and fetched page contents to the ICAP server for processing. In the ICAP server we can run anti-phishing filters against the URLs, page content, or user context. The filters can be based on well-known

anti-phishing tests used in browser toolbars or search engine result filters. These anti-phishing filters are written as plugins to the ICAP server and can be composed arbitrarily. We can even allow the user to choose which checks or transformations be made to the pages eventually delivered to the user.

Deploying a web-based proxy gives users the flexibility of choosing whether to filter their Web destinations. For sites that pose minimal risk to the loss of the user's credentials, or if users wish to bypass the proxy, it is unnecessary for those URLs to undergo the same processing as more important sites for banking or shopping.

However, a web based anti-phishing proxy brings up other issues. Users are taught that one defense against phishing attacks is to always manually enter the URL or to follow a bookmark. Using a proxy may train users to trust filtering their traffic through a third-party site. This could compromise the user's privacy. Attackers can easily set up what they claim to be a phishing-filter proxy to gather information about the user's accounts or simply to phish. Users will need to be able to authenticate the identity of the proxy. Another issue, unique to consumer mobile devices, is the design problem of being unobtrusive yet informative on devices with limited screen real estate.

6 Related Work

Previous studies investigated the causes for users falling for phishing attacks. Dhamija [12] studied user interface elements of web pages and browsers and identified errors users made when misidentifing fraudulent websites. Users lacked understanding of the difference between page content and browser program elements (chrome). They did not understand or failed to noticed the presence or absence of security indicators. We constructed similar spoofs of websites with changes to elements such as the faked browser chrome and show the increased susceptibility of users on limited browsers. Jakobsson [15] introduced "context aware" attacks where phishing attacks took into account factors surrounding a users access of a website. Customizing phishing sites for new non-traditional browser platforms can take advantage of increased user unfamiliarity to cause users to make security mistakes.

Anti-phishing tools try aid users in discriminating legitimate sites from phishing and are integrated into the browser to help stop phishing attacks. Spoofguard [11] identifies potential phishing sites via a series of heuristic tests examining the URL and page contents for traits found in known phishing sites. It communicates a warning to the user with an extra icon toolbar. Similar toolbars such as Spoofstick [8], Netcraft [7] Toolbar, and Trustbar [9] try to present similar warning cues to the user. However, Wu found that users fail to pay attention to these passive hints [16]. Egelman found that users do pay more attention when their focus is interrupted with active warnings and were less likely to be fooled [13]. Both Microsoft Internet Explorer 7 [6] and Mozilla Firefox 2 browsers [3], provide these active warning for confirmed phishing websites. While non-desktop browsers currently can not make use of these anti-phishing tools, this influenced our proxy design to help protect these browsers.

While it is difficult to have users to identity and react appropriately to phishing sites, automated techniques can identify phishing sites with high accuracy. Using the most relevant terms found in a pages content, CANTINA [17] identifies phishing sites when their search engine ranking is not sufficiently high. This method identified phishing sites with a 97% true positive rate. Garera [14] identified some common features in phishing sites such as URL obfuscation techniques and page lifetime and ranking from a search engine perspective. Simply classifying sites based on 15 features, phishing sites were identified with an average accuracy of over 97%.

7 Conclusions

We were able to identify several vulnerabilities on the iPhone's browser in its URL display, chrome, and input. From our user study, we observed that the majority of users without security backgrounds were unable to detect phishing attacks, even those familiar with the iPhone. When presented with the `scrollTo()` page and a fake chrome that made the page scroll incorrectly, users proceeded because they thought it was an iPhone glitch.

Porting the traditional browser to a mobile device with limited computation power requires considerably more foresight because these devices cannot upgrade to new versions as easily as desktops or laptops. The immediate design priority for usability diminishes the focus on security, but users are nevertheless being encouraged to do more of their important tasks from mobile devices, as evidenced by the creation of mobile sites by banking institutions, online merchants, and social networking sites. To diminish the power of malicious denizens on the Internet, we proposed a browser and device independent proxy service that may help bridge the gap between mobile and traditional browsers.

8 Acknowledgements

We wish to thank the anonymous reviewers who provided us with valuable feedback.

References

[1] Amazon. http://www.amazon.com/phone/.

[2] Bank of America Mobile Banking. https://www.bankofamerica.com/mobile/.

[3] Firefox Phishing Protection. http://www.mozilla.com/en-US/firefox/phishing-protection/.

[4] Internet Content Adaptation Protocol (ICAP). http://www.rfc-editor.org/rfc/rfc3507.txt.

[5] iPhone User's Guide. http://manuals.info.apple.com/en/iPhone_User_Guide.pdf.

[6] Microsoft Phishing Filter. http://www.microsoft.com/protect/products/yourself/phishingfilter.mspx.

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

[8] Spoofstick. http://www.spoofstick.com/.

[9] Trustbar. http://trustbar.mozdev.org/.

[10] W3Schools Browser Display Statistics. http://www.w3schools.com/browsers/browsers_display.asp.

[11] CHOU, N., LEDESMA, R., TERAGUCHI, Y., BONEH, D., AND MITCHELL, J. C. Client-side defense against web-based identity theft. In *NDSS '04: Proceedings of the 11th Annual Network and Distributed System Security Symposium* (February 2004).

[12] DHAMIJA, R., TYGAR, J. D., AND HEARST, M. Why phishing works. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems* (New York, NY, USA, 2006), ACM Press, pp. 581–590.

[13] EGELMAN, S., CRANOR, L. F., AND HONG, J. You've been warned: An empirical study of the effectiveness of web browser phishing warnings. In *CHI '08: Proceedings of the SIGCHI conference on Human Factors in computing systems* (New York, NY, USA, 2008), ACM Press.

[14] GARERA, S., PROVOS, N., CHEW, M., AND RUBIN, A. D. A framework for detection and measurement of phishing attacks. In *WORM '07: Proceedings of the 2007 ACM workshop on Recurring malcode* (New York, NY, USA, 2007), ACM, pp. 1–8.

[15] JAKOBSSON, M. Modeling and preventing phishing attacks. In *Financial Cryptography* (2005), p. 89.

[16] WU, M., MILLER, R. C., AND GARFINKEL, S. L. Do security toolbars actually prevent phishing attacks? In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems* (New York, NY, USA, 2006), ACM Press, pp. 601–610.

[17] ZHANG, Y., HONG, J. I., AND CRANOR, L. F. Cantina: a content-based approach to detecting phishing web sites. In *WWW '07: Proceedings of the 16th international conference on World Wide Web* (New York, NY, USA, 2007), ACM, pp. 639–648.