An Empirical Evaluation of Security Indicators in Mobile Web Browsers

Chaitrali Amrutkar and Patrick Traynor
Georgia Tech Information Security Center (GTISC)
Georgia Institute of Technology
chaitrali, traynor.cc @gatech.edu

Paul C. van Oorschot
School of Computer Science
Carleton University, Ottawa
paulv@scs.carleton.ca

Abstract

Mobile browsers are increasingly being relied upon to perform security sensitive operations. Like their desktop counterparts, these applications can enable SSL/TLS to provide strong security guarantees for communications over the web. However, the drastic reduction in screen size and the accompanying reorganization of screen real estate significantly changes the use and consistency of the security indicators and certificate information that alert users of site identity and the presence of strong cryptographic algorithms. In this paper, we perform the first measurement of the state of critical security indicators in mobile browsers. We evaluate ten mobile and two tablet browsers, representing over 90% of the market share, using the recommended guidelines for web user interface to convey security set forth by the World Wide Web Consortium (W3C). While desktop browsers follow the majority of guidelines, our analysis shows that mobile browsers fall significantly short. We also observe notable inconsistencies across mobile browsers when such mechanisms actually are implemented. Finally, we use this evidence to argue that the combination of reduced screen space and an independent selection of security indicators not only make it difficult for experts to determine the security standing of mobile browsers, but actually make mobile browsing more dangerous for average users as they provide a false sense of security.

1 Introduction

Mobile browsers provide a rich set of features that often rival their desktop counterparts. From support for Javascript and access to location information to the ability for third-party applications to render content through WebViews, browsers are beginning to serve as one of the critical enablers of modern mobile computing. Such functionality, in combination with the near universal implementation of strong cryptographic tools including SSL/TLS, allows users to become increasingly reliant upon mobile devices to enable sensitive personal, social and financial exchanges.

In spite of the availability of SSL/TLS, mobile users are regularly becoming the target of malicious behavior. A 2011 report indicates that mobile users are three times more likely to access phishing websites than desktop users [19]. Security indicators (i.e., certificate information, lock icons, cipher selection, etc.) in web browsers offer one of the few defenses against such attacks. A user can view different security indicators and related certificate information presented by the browser to offer signals or clues about the credibility of a website. Although mobile and tablet browsers appear to support similar security indicators when compared to desktop browsers, the reasons behind the increasing number of attacks on mobile browsers are not immediately clear.

In this paper, we perform the first comprehensive empirical evaluation of security indicators in mobile web browsers. The goal of this work is *not* to determine if average users take advantage of such cues, but instead to demonstrate that such indicators are lacking and thus fail to provide sufficient information for even experts. We believe that this distinction is critical because it highlights areas where not even the best trained users will be able to differentiate between malicious and benign behavior. Rather than an ad hoc analysis, we base our study on the recommendations set forward by the W3C for user interface security [10] as a proxy for best practices. In particular, we systematically measure which browsers strictly conform to the absolute requirements and prohibitions of this document. We perform our analysis across ten mobile and two tablet browsers, representing greater than 90% of the mobile market share [15], and then compare our results against the five most popular desktop browsers.

Our study makes the following contributions:

- **Widespread failure to implement recommended**

security indicators in mobile browsers: We perform the first *systematic* and *comprehensive* comparison between mobile, tablet and traditional browsers based on the best practices set forth in the W3C guidelines [10]. Whereas desktop browsers largely conform to these guidelines, mobile and tablet browsers fail to do so in numerous instances. We believe that this makes even expert users subject to attacks including an undetectable man-in-the-middle.

- **Mobile security indicators are implemented inconsistently across browsers:** Our study observed tremendous inconsistency in the presentation and availability of such indicators in mobile and tablet browsers, in contrast to traditional desktop browsers. Accordingly, many of the clues experts instruct average users to look for can no longer reliably be found on these platforms.

- **Newest security indicator is largely absent:** Extended Validation (EV) SSL indicators and certificates [9,28,35] were designed to improve assurance of the identity of the certificate holder. While this mechanism is not a requirement of the W3C recommendations, its use is pervasive in desktop browsers and virtually non-existent in mobile browsers. Mobile users are therefore unable to determine if certificates have undergone so-called extended validation, and sites using these certificates may be unable to justify their significant monetary investment in them.

Our measurements and observations from examining the most widely used mobile browsers lead us to make a number of bold assertions *which we expect will be viewed as controversial*, and thus worthy of discussion within the community. (1) Browser designers have been forced by the dramatic reduction in screen space to sacrifice a number of visual security features. The determination of which features are the most useful appears to have been by independent processes, as reflected in the different subsets of security indicators implemented across the mobile platforms. (2) Previous studies have overwhelmingly demonstrated that average users simply do not understand security indicators even on desktop browsers [22–24, 26, 37, 39]. Our measurements demonstrate that the display of security indicators on mobile platforms are considerably worse, to the extent that we as experts cannot express confidence in having sufficient information to take proper decisions. Consequently, we assert that the role of security indicators in mobile browsers offers little more than a false sense of security. The security user interface must therefore either be dramatically improved, to provide indicators of demonstrable use, or should be considerably simplified, to remove unusable, unreliable, or misleading artifacts. (3) We argue that the current practice of repeatedly forcing a user-base that is largely security un-savvy to make subtle security decisions is a losing game. Minor tweaks to the wordings of certificate interface dialogues, for example, may reach a slightly higher local maxima in terms of security improvements, but are highly unlikely to attain a more global maxima offering demonstrably better security. Given the real estate constraints of the increasingly dominant mobile platforms, our evidence shows that this current practice has actually resulted in a decrease in overall security signaling. Consequently, we raise questions not only about the viability of Extended Validation (EV) SSL certificates, but about the ongoing viability of SSL indicators themselves due to the inability to convey accurate, reliable information to users as necessary for subtle security decisions.

For these reasons, we believe that a measurement study as reported in the current paper is a requisite first step for our community to address these difficult issues and deal with these problems head-on.

The remainder of our paper is organized as follows: Section 2 explains the mandatory elements of the W3C guidelines; Section 3 provides the primary results of our evaluation; Section 4 presents ways in that a user can be misled about the identity of a website or the use of encryption and attacks that are enabled by this confusion; Section 5 discusses the presence of EV-SSL, the newest security indicator; Section 6 provides a discussion of our findings; Section 7 presents an overview of related research; and Section 8 offers concluding remarks.

2 Background on W3C Recommendations

The World Wide Web Consortium (W3C) has defined user interface guidelines [10] for the presentation and communication of web security context information to end-users of both desktop and mobile browsers. We provide a brief explanation of the W3C guidelines referenced within this paper. For clarity of terminology used in the guidelines and the later sections, **we encourage readers to refer to the definitions provided in the Appendix A.1**.

We chose a subset of the absolute requirements (MUST) and prohibitions (MUST NOT) specified in the W3C user interface guidelines.[1] We omitted the guidelines represented by clauses including the MAY, MAY NOT, SHOULD and SHOULD NOT keywords as they represent the optional guidelines [3]. We classify the

[1]The guidelines deemed to be the most critical, definitively testable and enabling attacks when violated were selected based on the authors' experience and knowledge of the area of SSL indicators.

Category	Browser Name	Version	Rendering Engine	Operating System	Device
Mobile	Android	2.3.3	Webkit	Android 2.3.3	Nexus One
	Blackberry	5.0.0	Mango	Blackberry OS 5.0.0.732	Bold 9650
	Blackberry	6.0.0	Webkit	Blackberry OS 6	Torch 9800
	Chrome Beta	0.16.4130.199	Webkit	Android 4.0.3	Nexus S
	Firefox Mobile	4 Beta 3	Gecko	Android 2.3.3	Nexus One
	Internet Explorer Mobile	*	Trident	Windows Phone 7.0.7004.0 OS	LG-C900
	Nokia Browser	7.4.2.6	Webkit	Symbian Belle	Nokia 701
	Opera Mini	6.0.24556	Presto	Android 2.3.3	Nexus One
		5.0.019802	Presto	iOS 4.1 (8B117)	iPhone
	Opera Mobile	11.00	Presto	Android 2.3.3	Nexus One
	Safari	*	Webkit	iOS 4.1 (8B117)	iPhone
Tablet	Android	*	Webkit	Android 3.1	Samsung Galaxy
	Safari	*	Webkit	iOS 4.3.5 (8L1)	iPad 2
Desktop	Chrome	15.0.874.106	Webkit	OS X 10.6.8	–
	Firefox	7.0.1	Gecko	OS X 10.6.8	–
	Internet Explorer	8.0.7600.16385	Trident	Windows 7	–
	Opera	11.52	Presto	OS X 10.6.8	–
	Safari	5.1.1	Webkit	OS X 10.6.8	–

Table 1: Details of the browsers used for experimental evaluation. We selected Android browser version 2.3.3 since it has the highest market share among the available Android platforms [17]. (*: The version numbers of these browsers were not apparent. We have used the default browsers shipped with the referenced version of the OS.)

W3C guidelines into *five* categories: identity signal, certificates, TLS indicators, robustness and error messages.

1) Identity signal: availability:

The security indicators showing identity of a website MUST be available to the user either through the primary or the secondary interface at all times.

2) Certificates: required content:

In addition to the identity signal, the web browsers MUST make the following security context information available through information sources (certificates): the webpage's domain name and the reason why the displayed information is trusted (or not).

3) TLS indicators:

a) Significance of presence: Any UI indicator (such as the padlock) MUST NOT signal the presence of a certificate unless all parts of the webpage are loaded from servers presenting at least validated certificates over strongly TLS-protected interactions.

b) Content and Indicator Proximity: Content MUST NOT be displayed in a manner that confuses hosted content and browser chrome[2] indicators, by allowing that content to mimic chrome indicators in a position close to them.

c) Availability: The TLS indicators MUST be available to the user through the primary or the secondary interface at all times.

[2]A browser chrome refers to the graphical elements of a web browser that do not include web content.

4) Robustness: visibility of indicators:

Web content MUST NOT obscure the security user interface.

5) Error messages:

a) Interruption: Both warning/caution and danger messages MUST interrupt the user's current task, such that the user has to acknowledge the message.

b) Proceeding options: Warning/caution messages MUST provide the user with distinct options for how to proceed (i.e., these messages MUST NOT lead to a situation in which the only option presented to the user is to dismiss the warning and continue).

c) Inhibit interaction: The interactions for danger messages MUST be presented in a way that makes it impossible for the user to go to or interact with the destination website that caused the danger situation to occur, without first explicitly interacting with the danger message.

3 Empirical Observations

We evaluate ten mobile and two tablet browsers against the W3C recommended practices for security indicators. The details of the browsers are provided in Table 1. For each of the guidelines described in Section 2, we create and run a set of experiments to verify compliance on all the candidate browsers and record our observations. All the experiments were performed on web browsers on

Mobile and Tablet Browsers (See Table 1 for versions)	1) Identity signal: availability		2) Certificates: required content	
	Owner information available?	Certificate issuer's information available?	Domain name available?	Information on why certificate trusted available?
Android	•	•	•	×
Blackberry Mango	•	•	•	•
Blackberry Webkit	•	•	•	•
Chrome Beta	•	•	•	×
Firefox Mobile	•	•	•	×
iPhone Safari	×	×	×	×
Nokia Browser	•	•	•	×
Opera Mini	×	×	×	×
Opera Mobile	×	×	×	×
Windows IE Mobile	×	×	×	×
Safari on iPad 2	×	×	×	×
Android on Galaxy	•	•	•	×

Table 2: Results of experiments on candidate mobile browsers to test compliance with the first two W3C guidelines given in Section 2. Each guideline column consists of sub-columns stating the experiments performed on the browsers. A × implies that the browser does not comply with the respective W3C guideline. A • implies that the browser complies with the respective W3C guideline. Note that all the desktop browsers are compliant to the same guidelines.

real mobile phones, and are recreated in the respective emulators to generate many of the figures throughout the paper. The browser versions used in our evaluation are approximately the latest as of February 12^{th}, 2012. Tables 2 through 6 provide the synopsis of the results of our experiments.

3.1 Identity Signal: Availability

An identity signal contains information about the owner of a website and the corresponding certificate issuer. Before issuing a certificate, the certificate provider requests the contact email address for the website from a public domain name registrar, and checks that published address against the email address supplied in the certificate request. Therefore, the owner of a website is someone in contact with the person who registered the domain name. Popular browsers represent the owner information of a website using different terminology including owner, subject, holder and organization.

We visited a public webpage presenting a trusted root certificate from all the candidate browsers. We then evaluated the browsers for the presence of identity signal, either on the primary or the secondary interface.

Observations: The IE Mobile, iPhone and iPad Safari, and Opera Mini and Mobile browsers do not provide a user interface to view certificates. Accordingly, none of these five browsers comply with the W3C recommendations We note that when a website presents a certificate that is from a CA not from a trusted root, all the browsers provide an interface to view the certificate via an error message. The Android mobile and tablet, Blackberry Mango and Webkit, Chrome Beta and Nokia browsers always allow a user to view certificates (both trusted and untrusted) and therefore comply with this guideline. A user is required to click the lock icon to view certificate information on the Chrome Beta and Blackberry Mango browsers. However, the browsers do not provide any visual indication to the user about this process of accessing the certificate information. Browsers supporting a UI for viewing certificate information provide a clear indication in the "options" in the browser menu. Although the Firefox Mobile browser does not support a certificate UI, it displays the identity information of a website when the site identity button is clicked. All desktop browsers comply with this guideline. Table 2 provides the summary of our results.

3.2 Certificates: Required Content

In addition to the identity signal content, a certificate from a website must provide the same website's domain name and the reason why the displayed information is trusted (or not). Trust reasons include whether or not a certificate was accepted interactively, whether a self-signed certificate was used, whether the self-signed certificate was pinned to the site that the user interacts with, and whether trust relevant settings of the user agent were otherwise overridden through user action. We believe that information such as "certificate is implicitly trusted" and "the certificate chain is trusted/valid" also conveys the reason behind a browser trusting or not trusting a particular website.

We analyzed the candidate browsers for the presence of the required certificate content by visiting a website that uses strongly TLS-protected connection with its clients.

Observations: The IE Mobile, iPhone and iPad Safari, and Opera Mini and Mobile browsers do not provide a user interface to view certificates from trusted CAs. Therefore, these browsers fail to meet the W3C guideline. Additionally, even though the remaining mobile and tablet browsers provide a user interface to view certificate information, they do not provide an explanation on why a particular certificate is trusted. Only the Blackberry Mango and Webkit browsers comply with the guideline by making all the required parts of a certificate available. When a website presents a certificate from a trusted CA, the Blackberry Mango and Webkit browsers show the reason "certificate is implicitly trusted". Therefore, all but two mobile and tablet browsers fail to meet this W3C guideline. All desktop browsers follow this guideline correctly. Table 2 provides the summary of our results.

3.3 TLS Indicators

TLS indicators include the `https` prefix, the padlock icon, information about the ciphers used in the connection and URL coloring (or site identity button) to depict the difference between EV-SSL and SSL certified webpages.

a) Significance of presence: If a web browser displays a TLS indicator for the presence of a certificate for a webpage consisting of content obtained over both `http` and `https` connections (mixed content), this guideline is not followed.

We created a simple webpage that uses a strong TLS connection to retrieve the top-level resource and embedded a map obtained from a third-party over an unsecured `http` connection. We rendered this webpage on the candidate browsers and analyzed the browsers for the presence of two basic TLS security indicators: the `https` URL prefix and the padlock icon. If a browser shows any of these two indicators on a mixed content webpage, it does not follow the W3C guideline.

Observations: The Blackberry Mango, Blackberry Webkit and IE Mobile browsers display a lock icon on a webpage holding mixed content, thus failing to meet the W3C guideline. The Blackberry Webkit and IE Mobile browsers display a mixed-content warning and, if the user proceeds to the webpage, a lock icon is displayed. The Android browsers on the mobile and tablet devices present an open lock with a question mark inside the lock. The Chrome Beta browser displays a closed lock with a cross on top and a striked through `https` URL prefix for a mixed content webpage. This behavior of Android and Chrome is inconsistent with the other browsers. Therefore, it is necessary for the users of these browsers to understand the meaning of the new symbols in order to interpret its reference to mixed content on a webpage.

All browsers display the `https` URL prefix either on the primary or the secondary interface. We note that this issue is present even in popular desktop browsers. The behavior of displaying the `https` URL prefix on a mixed content webpage fails to meet the W3C recommendation in both the desktop and mobile environments as shown in Tables 3 and 4.

b) Content and Indicator Proximity: The padlock icon used as a security indicator and the favicon used as an identity element of a website are two popular elements that use a browser's chrome. If a browser allows a favicon to be placed next to the padlock, an attacker can feign a secure website by mimicking the favicon as a security indicator. We evaluated this scenario by visiting a webpage over a strong TLS connection from all candidate browsers and observing the relative locations of the favicon and padlock.

Observations: The Android mobile browser does not follow the W3C guideline. The browser places the favicon of a webpage beside the padlock icon. All other browsers adhere to this guideline, as shown in Tables 3 and 4.

We observed several inconsistencies in the use and position of the padlock icon and the favicon in the mobile and tablet browsers. As shown in Figure 1, the favicon is displayed only on the Android (mobile and tablet), Blackberry Webkit and Firefox Mobile browsers. The remaining mobile and tablet browsers never display a favicon. This behavior is inconsistent with desktop browsers. We believe lack of screen space to be one of the drivers behind the removal of the favicon from the mobile environment. In addition to the almost total lack of use of favicons, we also noticed that the position of the padlock icon in mobile browsers is inconsistent across different mobile browsers. In the past, researchers have shown that the padlock icon is the security indicator most often noticed by users [24, 39]. Traditional desktop browsers generally display the padlock icon in the address bar. However, all mobile and tablet browsers except Android (mobile and tablet), Blackberry Webkit, Chrome Beta and IE Mobile browsers display the lock icon on the title bar instead of the address bar. We believe that the reason behind this shift of location of the padlock icon in the mobile and tablet browsers is the non-persistent availability of the address bar to the user. Whenever a user starts interacting with a webpage, most mobile browsers hide the address bar to accommodate more content on the small screen.

c) Availability: We studied the presence of the lock icon,

Mobile and Tablet Browsers (See Table 1 for versions)	TLS indicators					
	3a) significance of presence		3b) position	3c) availability		
	Mixed content: no lock shown?	Mixed content: no `https` shown?	Favicon not next to lock icon?	`https` prefix available?	Lock shown?	Cipher details available?
Android	Open lock with a question mark	×	×	• (*s*)	•	×
Blackberry Mango	×	×	•	• (*s*)	•	•
Blackberry Webkit	×	×	•	• (*s*)	•	•
Chrome Beta	Closed lock with a cross on top	https striked through	•	• (*s*)	•	•
Firefox Mobile	No security indicators shown	×	•	• (*s*)	• On clicking the site identity button	×
iPhone Safari	•	×	•	• (*s*)	•	×
Nokia Browser	•	×	•	• (*s*)	•	×
Opera Mini	•	×	•	• (*s*)	•	×
Opera Mobile	•	×	•	• (*s*)	•	×
Windows IE Mobile	×	×	•	• (*s*)	•	×
Safari on iPad 2	•	×	•	• (*s*)	•	×
Android on Galaxy	Open lock with a question mark	×	•	• (*s*)	•	×

Table 3: Results of experiments on candidate mobile browsers to test compliance with the W3C guidelines 3a, 3b, and 3c given in Section 2. The symbol notation is as defined in Table 2. 's': Implies that the `https` URL prefix is present on the 's'econdary interface.

Desktop Browsers (See Table 1 for versions)	TLS indicators					
	3a) significance of presence		3b) position	3c) availability		
	Mixed content: no lock shown?	Mixed content: no `https` shown?	Favicon not next to lock icon?	`https` prefix available?	Lock shown?	Cipher details available?
Chrome	Lock with a yellow triangle	×	•	• (*p*)	•	•
Firefox	•	×	•	• (*p*)	• On clicking the site identity button	•
IE	•	×	•	• (*p*)	•	×
Opera	•	×	•	• (*p*)	•	•
Safari	•	×	•	• (*p*)	•	×

Table 4: Results of experiments on desktop browsers to test compliance with the guidelines in Table 3. The symbol notation is as defined in Table 2. 'p': Implies that the `https` URL prefix is present on the 'p'rimary interface.

the `https` URL prefix and details of the cipher used in a TLS connection by visiting a TLS protected webpage using all candidate browsers. The padlock icon and the `https` URL prefix are primary interface indicators and cipher information is a secondary interface indicator on desktop browsers.

Observations: Websites handling sensitive digital transactions (such as banks) ask users to search for the `https` URL prefix to ensure security of their transactions. Therefore, easy access to the `https` URL prefix is important. This indicator is present in the address bar (primary interface) of desktop browsers and is clearly visible to the user at all times. Among the mobile and tablet browsers, all but the Blackberry Mango browser display the `https` URL prefix in the address bar. The Blackberry Mango browser does not have an address bar and provides a choice to view the webpage's URL from the browser's options. This setting requires a user to be knowledgeable of the change to be able to find the URL of the current webpage and also makes the `https` URL prefix a secondary interface indicator. Although the other mobile browsers display the `https` URL prefix in the address bar, they hide the address bar (except Chrome Beta) for better usability. In the Chrome Beta browser, if the URL of a webpage is longer than the screen size, the `https` URL prefix is hidden. Since a user is required to interact with the address bar to view the URL prefix of a webpage, the `https` URL prefix becomes a secondary interface indicator in all mobile and tablet browsers. This increases the likelihood of a successful

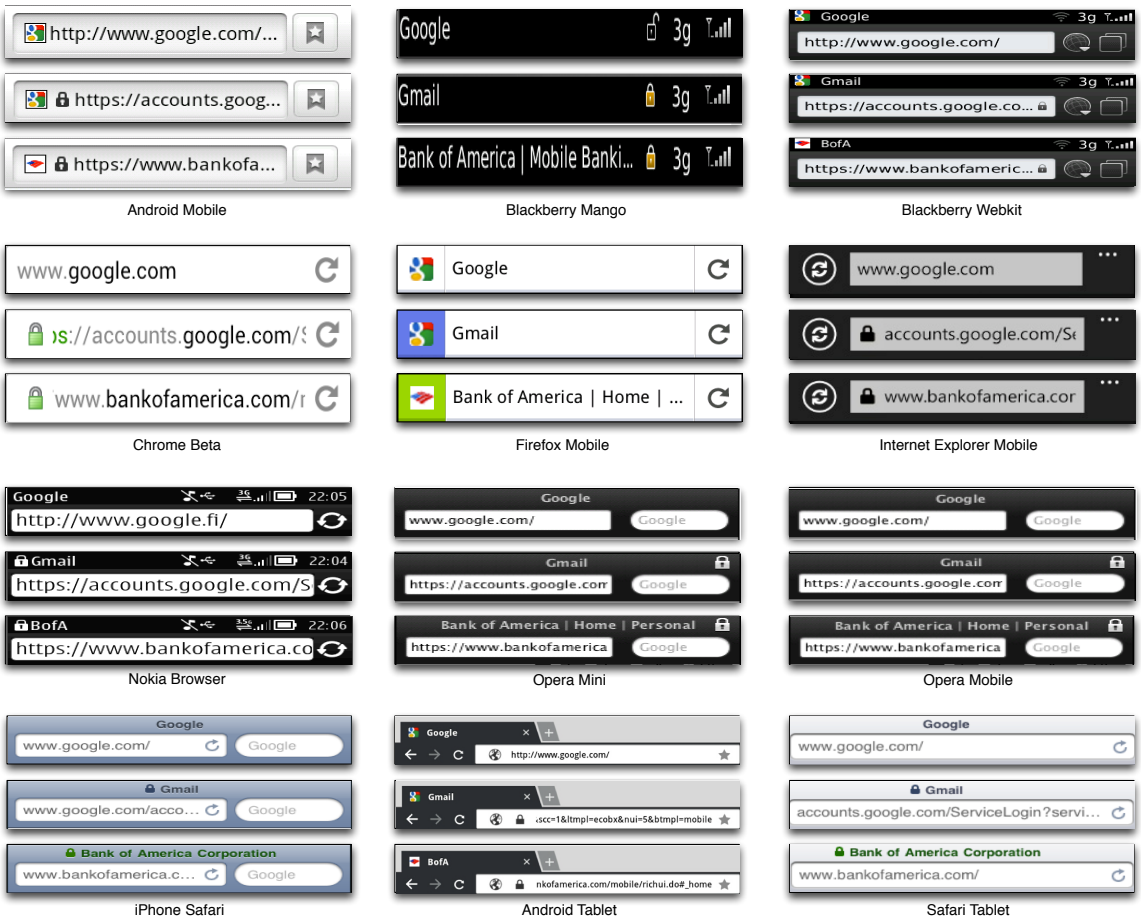

Figure 1: Security indicators on the primary interface (address bar) of all the mobile and tablet browsers. Every browser has three screenshots of the address bar: from top to bottom, the websites are Google over an `http` connection, Gmail over a secure connection with an SSL certificate and Bank of America over a secure connection with an EV-SSL certificate.

downgrade attack (e.g., SSLstrip [7] attack) on the mobile and tablet browsers, since a user requires effort to view the `https` URL prefix.

The information about the ciphers used in setting up the TLS connection between a website server and the user's browser is not available in any of the browsers except Blackberry Mango and Webkit and Chrome Beta. Accordingly, all the mobile and tablet browsers except three do not comply this W3C guideline for our experiments. Tables 3 and 4 provide the summary of our results.

3.4 Robustness: Visibility of Indicators

The TLS indicators generally found on the primary interface are lock icon, `https` URL prefix, URL coloring and site identity button. Typically, the address bar in a web browser holds these indicators. Therefore, we examined whether web content overwrites or pushes the address bar containing security indicators out of a user's view during browsing.

Observations: Presumably, in order to free up screen real estate for other purposes, the address bar on all but two mobile and tablet browsers is overwritten by web content once a webpage is rendered and/or when a user starts interacting with the page. The IE Mobile browser always displays the address bar in the portrait view. However, the address bar is never displayed in IE Mobile when a user interacts with a webpage in the landscape mode. The Chrome beta browser makes the address bar persistently available in both the portrait and landscape modes. Out of the two tablet browsers, only the tablet Safari browser avoids the security indicators on the address bar being overwritten by a webpage's content, therefore allowing a persistent view of the security indicators on the primary interface. The Android tablet browser hides the address bar once a webpage is ren-

Mobile and Tablet Browsers (See Table 1 for versions)	**4) Robustness** Content does not obscure indicators on the address bar?	**5) Error messages**			
		5a) Interruption		**5b) Proceeding options** (for warnings)	**5c) Inhibit interaction** (for danger messages)
		Warning (mixed content)	Danger (self-signed cert)		
Android	×	×*	•	*NA**	•
Blackberry Mango	NA	×*	•	*NA**	•
Blackberry Webkit	×	•	•	"Continue, Close connection, View cert, Trust cert" options	•
Chrome Beta	•	×*	•	*NA**	•
Firefox Mobile	×	×*	•	*NA**	•
iPhone Safari	×	×*	•	*NA**	•
Nokia Browser	×	•	•	•	•
Opera Mini	×	×*	×	*NA**	×
Opera Mobile	×	×*	•	*NA**	•
Windows IE Mobile	×	•	•	"Yes and No" options	•
Safari on iPad 2	•	×*	•	*NA**	•
Android on Galaxy	×	•	•	"Continue, View Certificate, Go Back" options	•

Table 5: Results of experiments on mobile and tablet browsers to test compliance with the W3C guidelines 4, 5a, 5b and 5c given in Section 2. The symbol notation is as defined in Table 2. *NA*: Implies that the concerned experiment is not applicable to that browser, the reasoning can be found in the text. (*: Our view is that a browser should display a warning message for a webpage holding mixed content, to avoid misleading users trained to interpret SSL indicators to mean that the (entire) webpage is secured.) $\times^*$: Implies that the browser fails to warn a user according to our view.

Desktop Browsers (See Table 1 for versions)	**4) Robustness** Content does not obscure indicators on the address bar?	**5) Error messages**			
		5a) Interruption		**5b) Proceeding options** (for warnings)	**5c) Inhibit interaction** (for danger messages)
		Warning (mixed content)	Danger (self-signed cert)		
Chrome	•	×* site identity button shows a warning	•	*NA**	•
Firefox	•	×*	•	*NA**	•
IE	•	•	•	"Yes, No" More info options	•
Opera	•	×*	•	*NA**	•
Safari	•	×*	•	*NA**	•

Table 6: Results of experiments on desktop browsers to test compliance with the same guidelines as Table 5. The symbol notation is as defined in Table 2 and Table 5.

dered. Tables 5 and 6 show that all the candidate desktop browsers follow this guideline unlike the mobile and tablet browsers.

3.5 Error Messages

We created example scenarios that demand the warning/caution and danger messages, given the definitions in the W3C document. The W3C document provides examples of scenarios that demand a danger alert. However, as the document does not specify any scenarios that should trigger warnings, we carried out our tests using the following scenario.

We classified the scenario of a browser rendering a mixed content webpage as one that should trigger a warning. This is because on a webpage with both insecure and secure content, the user may or may not interact with the insecure content on the webpage. Therefore, the browser system is unable to positively determine whether the user is at risk. In contrast, we used an example scenario given in the W3C document for our experiments on danger messages. The W3C document defines 'rendering a webpage presenting a self-signed certificate' as one that should trigger a danger message, since the certificate is not from a trusted root.

a) Interruption: We examined whether the mobile and tablet browsers display a warning or danger message in our example scenarios. We further observed the nature of the messages to confirm that they actually interrupt the user's actions as specified by the W3C guidelines and are not displayed at a position on the screen which a user can ignore and continue interacting with the website.

Observations: Only four mobile and tablet browsers (Android tablet, Blackberry Webkit, IE Mobile and Nokia) interrupt the user by displaying a warning about the existence of insecure content on a mixed content webpage, before the webpage is rendered. The other browsers do not interrupt the user by displaying a warning. The iPhone Safari browser shows a mixed content warning on a console that needs to be enabled by a user and is intended for developers. We believe that most iPhone Safari users are unlikely to enable the debug console, carefully browse the debug messages and therefore understand the presence of mixed content. The Chrome Beta browser shows a warning about mixed content when the crossed padlock in the address bar is clicked. Among the desktop browsers, only IE displays a mixed content warning, thereby interrupting a user.

All but one of the mobile and tablet browsers comply with the interruption guideline by displaying a danger message, when a webpage with a self-signed certificate is rendered. The Opera Mini browser is the only browser that does not display a danger message in this scenario; it simply renders the webpage and does not show any TLS indicators.

b) Proceeding options: We examined whether the warning message displayed for a mixed content webpage provides a user with more than one option to proceed after interruption.

Observations: Only the Android tablet, Blackberry Webkit, IE Mobile and Nokia browsers display a warning message when navigated to a mixed content webpage. The IE Mobile browser informs the user about the presence of unsecured content on the webpage and provides two options for continuing: <Yes, No>. However, there is no option to the user to view the certificate provided by the top-level website using a secured connection. Conversely, the Android tablet and Nokia browsers provide an option to view a website's certificate. The options presented by the Android tablet browser are <Continue, View Certificate, Go back>, where the "Go back" option navigates the user to a webpage viewed right before the mixed content webpage. The options provided by the Nokia browser are <Accept this time only, Accept permanently, Certificate details>. The Blackberry Webkit browser provides the options to <Continue, Close Connection (default), View Certificate, Trust Certificate>. Among the desktop browsers, IE provides <Yes, No, More info> options to proceed when a mixed content webpage is rendered.

c) Inhibit interaction: This guideline requires a browser to prevent a user from interacting with a website that triggers a danger message, before user interaction with the danger message. We visited a website presenting an untrusted self-signed certificate from all the browsers.

Observations: All mobile and tablet browsers except Opera Mini display a danger message on receiving a self-signed certificate. Additionally, they restrict a user's interaction to the danger message. A user is unable to access the website content before explicitly interacting with the danger message. The Opera Mini browser does not show an error for self-signed certificates. It simply routes the user to the webpage presenting the self-signed certificate. The Chrome Beta browser displays a cross on the `https` URL prefix and the padlock icon even after a webpage with a danger message is loaded following user interaction. All desktop browsers correctly follow this guideline. Table 5 and Table 6 summarize our experimental results of the error message guidelines.

We discuss additional findings that are not directly related to the guidelines studied in this section in the Appendix A.2.

4 User Deception and Potential Attacks

The W3C user interface guidelines, which we use as a proxy for best practice, are an effort to communicate security information to users such that they can make informed decisions about websites that they visit. If these guidelines are not implemented by a browser, users are more easily misled about the identity of a website or the security of a connection. We discuss four attacks that are enabled on browsers violating one or more of the W3C guidelines. Table 7 provides a summary of potential attacks described in this section on the candidate browsers.

i) Phishing without SSL: An attacker masquerades as a trustworthy entity in a phishing attack. By closely imitating a legitimate website's identity information in combination with lock icon spoofing, a malicious website can launch a phishing attack without SSL on a browser violating the W3C guidelines 1, 2 and 3b as follows.

An attacker buys a domain name that closely resembles the domain name of the legitimate website. For example, to spoof `www.bankofamerica.com`, the attacker buys the domain name `www.bankofamericaa.com`. The attacker then imitates the content of the targeted legitimate website. Instead of spending money on purchasing an SSL certificate to increase the "false" credibility of the malicious website, an attacker instead makes the favicon of the malicious website a lock image.

Attacks	Android	Blackberry Mango	Blackberry Webkit	Chrome Beta	Firefox Mobile	iPhone Safari	Nokia	Opera Mini	Opera Mobile	IE Mobile	Safari on iPad2	Android on Galaxy
Phishing without SSL	×	•	•	•	•	•	•	•	•	•	•	•
Phishing with SSL	•	×	•	•	•	×	•	×	×	×	×	•
Phishing using a compromised CA	•	×	•	•	•	×	•	×	×	×	×	•
Industrial espionage/ Eavesdropping	×	×	×	×	×	×	×	×	×	×	×	×

Table 7: Summary of potential attacks on candidate mobile browsers. A × implies that the attack is possible. A • implies that the corresponding attack is not possible on the browser.

Therefore, the closely imitated domain name provides an impression of correct identity of the intended website and the spoofed lock provides an illusion of strong encryption.

When this malicious website is rendered in a browser that makes viewing the URL of the website difficult, situates the favicon next to the padlock icon and does not offer a UI to view identity information such as website owner's name, even an advanced user might be subjected to phishing.

ii) Phishing with SSL: Spoofing only the lock icon may not be adequate to launch a successful phishing attack. To increase the credibility of a phishing website, the attacker can buy an inexpensive SSL certificate for the website. The presence of a valid certificate causes a browser to display SSL indicators such as the `https` URL prefix and URL coloring (or colored site identity button) in addition to the lock icon in the browser's chrome. If a user blindly trusts just these SSL indicators and can not verify additional identity information of the website (violation of guideline 1 and 2), he can be subjected to a phishing attack.

iii) Phishing using a compromised CA: Compromising a CA allows an attacker to obtain rogue certificates for legitimate websites. There have been several such attacks recently [13, 14]. If a user's browser trusts a CA, the browser will accept all certificates signed by the CA without showing any warning to the user. This behavior persists even when the same CA is compromised and the necessary update to remove the trusted CA from the browser has not been installed. An expert user who is knowledgeable of a CA compromise can verify every certificate issuer's organization in the certificate chain, therefore declining interacting with a malicious website with a rogue certificate. If a browser fails to meet guidelines 1 and 2, thereby not presenting user interface to enable certificate viewing, even an expert user could be exposed to a phishing attack.

iv) Industrial espionage / eavesdropping: A man-in-the-middle (network) attacker can use any one of the cipher downgrade, substituting `http` for `https` or inserting mixed content techniques for user deception to launch an eavesdropping attack on a user's session as follows:

SSLstrip attack: The SSLstrip [7] man-in-the-middle attacker sits on a local network and intercepts traffic. When the attacker detects a request to an encrypted `https` site, he substitutes a duplicate of the intended destination as an unencrypted `http` site. This switching strips away the security that prevents a third party from stealing or modifying data, while deceiving the server that an encrypted page has been sent to the client. The network attacker can also fake a lock icon in the stripped `http` page, by replacing the favicon by a lock icon [30]. If the `https` prefix is not available to a user persistently, he may not be able to recognize that he is using an unsecured connection by noticing the change from `https` to `http` in the address bar. A browser not displaying the `https` prefix persistently does not follow requirement 3c in Section 2.

Cipher downgrade attack: A man-in-the-middle (network attacker) can tamper with the initial messages sent by a client browser to establish an SSL connection with a website server. Before a TLS connection is set up, a client and server exchange a list of ciphers that they support. A network attacker can modify the list of supported ciphers sent by the client to a list containing only weak ciphers, and then forward the client's request/response to the server. On receiving a list of only weak ciphers (e.g., DES-CBC-SHA), the server can either drop the connection because no ciphers are mutually supported, or provide support for that cipher and begin an encrypted session with the weak cipher. When a connection using the weak cipher is initiated, all the data in transit is protected using the weak cipher's encryption scheme. This allows a network attacker to capture the stream of data and break the weak encryption offline. The attack is also useful to mislead even an expert user that their transactions are over a connection with strong encryption algorithms, since the SSL indicators such as `https` URL prefix and lock icon are present even for a connection using a weak cipher. If a browser does not display cipher information, it fails to meet the W3C requirement 3c in Section 2.

	Android	Blackberry Mango	Blackberry Webkit	Chrome Beta	Firefox Mobile	iPhone Safari	Nokia	Opera Mini	Opera Mobile	IE Mobile	Safari on iPad2	Android on Galaxy
EV-SSL v/s SSL differentiation	×	×	×	×	•	•	×	×	×	×	•	×

Table 8: Results of whether browsers differentiate between EV-SSL and SSL certified webpages. A × implies that the browser does not provide differentiating indicators and a • implies the presence of such indicators in the browser.

Mixed content attack: A man-in-the-middle attacker can tamper (e.g., code injection) with the unencrypted content present on a webpage consisting of mixed content and replace the original content with any malicious content of his choice. If a web browser displays SSL indicators for a webpage containing mixed content (violation of guideline 3a), even an expert user may be unable to detect a network attack exploiting the mixed content on a webpage.

The results of our experiments combined with this threat model make the candidate mobile and tablet browsers susceptible to phishing and eavesdropping attacks as shown in Table 7.

5 Inadequate EV-SSL Differentiation

The W3C guidelines do not establish recommendations for the browser user interface to signify the difference between EV-SSL [9, 35] and SSL certificates. The sole distinction between an SSL and an EV-SSL certificate from a user's perspective is the set of indicators on his browser. For example, the Firefox desktop browser uses a green site identity button to convey the presence of an EV-SSL certificate on a website. However, the site identity button is blue in the same browser when a website with an SSL certificate is rendered.

SSL certificates can be 'domain-validation-only' with minimal verification performed on the details of the certificate. Since any successful SSL connection causes the padlock icon to appear, users are not likely to be aware of whether the website owner has been validated or not. Therefore, fraudulent websites have started using inexpensive domain-validated SSL certificates with minimal verification to gain user trust. EV-SSL certificates were created to restore confidence among users that a particular website owner has been subjected to more rigorous vetting than simply determining control of a domain name [38]. If browsers do not differentiate between SSL and EV-SSL certificates, then the fundamental motivation [9] behind EV-SSL certificates becomes void. So too does the incentive for site owners to pay extra for such certificates – an SSL certificate from Go Daddy costs $12.99/year [1] and an EV-SSL certificate from VeriSign costs $1499/year [2]. In a browser with no differentiation between SSL and EV-SSL, both these certificates appear the same from a user's perspective. An adversary holding a domain name and willing to spend money for the SSL certificate would then trigger exactly the same user interface elements to users, and thus appear to provide identical guarantees as a website certified by the more expensive certificate.

Experimental observations: We browsed both EV-SSL and SSL certified webpages using all the candidate browsers. With the exceptions of the Firefox Mobile and the iPhone and iPad Safari browsers, none of the mobile or tablet browsers display any indicators that differentiate between EV-SSL and SSL certified webpages. The Firefox Mobile browser changes the site identity button green or blue to depict the presence of EV-SSL and SSL certified webpages respectively. The Safari mobile and tablet browsers use green and blue coloring of the 'title' to represent the difference between EV-SSL and SSL. This behavior of the Firefox Mobile and the Safari browsers is consistent with their desktop counterparts. All tested desktop browsers indicate the difference between these two certificate types. Table 8 provides a summary of the results.

We note the following advice within official guidelines from the CA/Browser Forum [6]:

In cases where the relying application accepts both EV and non-EV certificates, it is recommended that the application's behavior differ in a distinct way for each type of certificate. Application developers should consider the EV treatment offered by other application developers that also recognize EV certificates and, where practical, provide consistent treatment.

We assert that much more specific advice is essential, for example, in a revision or extension of the W3C user interface guidelines [10] and potentially in the CA/Browser Forum's own recommendations, as well as in appropriate supporting standards.

6 Discussion and Implications

Having performed comprehensive measurements of security indicators in the most widely used mobile browsers (over 90% of the market share), we now discuss what we see to be the implications. We have separated this from our measurements to allow independent interpretations by others.

It appears quite clear that the lack of available screen real estate has dramatically influenced the use of security indicators on mobile devices. Whereas traditional

browsers have a considerably richer space budget to accommodate current (and even experimental) indicators, mobile browsers struggle to clearly show content, let alone signals of the origin and security of the connection to that content. Our measurements suggest that browser vendors have individually made different decisions to best balance these competing demands, independently selecting to implement different subsets of the indicators in use on desktop browsers. Unfortunately, our study also reveals that the choices made by each browser removes signals available to users to detect, and possibly avoid, specific attacks—even if only by expert users. It is *not* our conclusion that implementing exactly the same security indicators on mobile platforms is impossible from an engineering perspective; but rather, the real estate limit of mobile phones makes unclear the means of doing so in a manner that does not simply overwhelm the content on small mobile screens. As an example and as shown in Figure 1, adding the `https` indicator to the address bar on the primary interface would make the vast majority of the URLs even less readable.

Moreover, even if possible, it is entirely unclear that duplicating all desktop security indicators on mobile browsers is the best course of action to pursue. Studies over the last decade have repeatedly shown that *average* users either ignore or do not understand browser security indicators [22–24, 26, 37, 39]. While usability and HCI experts have contributed to some improvements, the community appears to be at a stalemate, lacking new ideas—even across 10 mobile browsers—to improve (or even retain) security signaling while preserving usability. For example, wording and sequencing of warning dialogs has arguably been improved, but such changes are minor enhancements at best—and appear to move us at best to local maxima, constrained by the current set of indicators and the SSL security framework, while higher global maxima seem beyond reach.

Given these observations, our work reaches the following logical conundrum: Mobile browsers are currently vulnerable to attacks that could potentially be ameliorated by implementing known security indicators at least as well as on desktops. These indicators have not been implemented due to space constraints. Moreover, were these indicators to be implemented on mobile browsers (likely at the expense of page content), it is entirely unclear that users would profitably notice them. Accordingly, our community faces a difficult question:

> *Should mobile browsers vendors continue to include only a subset of indicators and provide a false sense of security to even expert users, or implement the full corpus of indicators (which past studies indicate users will largely ignore) at the risk of drastically reducing the usability of these applications?*

Answering this question invokes a discussion of the limits an SSL security framework, based on third-party certificates, with trust deriving from a combination of browser configuration data and user input. A key assumption is that users can be provided with sufficient interface cues (e.g., regarding the origin and algorithms used to exchange sensitive content) and then take correct security actions based thereon. However, as the cited studies (and others) indicate, the largely security unsavvy average user cannot perform this task. Presented with subtle streams of information, users cannot realistically be expected to sort out differences between cryptographic algorithms, nor proper responses to increasing incidents of CA compromise [13, 14]. In short, continuing to "punt" security decisions to users who, in spite of browsing daily for the past decade have simply not become security experts, is a strategy that appears no longer to anyone's advantage. Compounded by the screen constraints of mobile devices, and these devices becoming the dominant means of end-user computing on the planet, continuing the current approach becomes even less plausible. We argue that something must change.

As a straw-man suggestion, to start conversation, we encourage discussion of the following resolution. As mobile users increasingly offload computationally expensive operations to "the cloud" (e.g., complex image rendering), so too might they outsource more security decisions. Large external organizations or specialist entities are more likely to have an understanding of the currently appropriate security footing for a browser (e.g., sufficiently strong algorithms, websites using certificates from bad CAs, etc.). Accordingly, the *average* user could be much better served by simply receiving a single indicator representing the opinion of the identity and security of the connection through one of these entities. Desktop browsers such as Chrome are already beginning to rely on such services to protect users from malicious downloads and phishing websites, and other researchers have proposed using CDNs to provide more expensive security services (e.g., defense against heap spraying attacks) [29]. Based on browser configuration, expert users could continue to make their own decisions, perhaps through secondary interfaces. In particular, given that our study shows that the majority of security information available in mobile browsers is in such secondary interfaces, expert users could look at a more complete set of information than currently available, and make more informed decisions themselves. This approach would both minimize the space required for security indicators and would potentially provide a more useful metric for users.

We do not believe that this straw-man proposal is without its own issues, but rather that the community is overdue to directly confront the (lack of) security provided by current indicators. Our measurement study shows that

currently, the value is clearly insufficient.

7 Related Work

TRADITIONAL BROWSER INDICATORS: Traditional desktop browsers contain a range of security indicators in the chrome of the browser including the lock icon, the `https` URL prefix, and certificates. Several studies have indicated that these security cues used in desktop browsers go unnoticed [23,24,34,35,39] or are absent in websites [36]. Although domain name mismatches between certificates and websites are observed often [38], Sunshine et al [37] showed that users ignore TLS warnings for domain name mismatches, and showed that users ignore TLS warnings for expired certificates and unknown CAs. Moreover, a majority do not understand these warnings. The lock icon is the security indicator most often noticed [24, 39]. However, even when used as a security cue by users, many do not fully understand its meaning [22–24] and its absence also often goes unnoticed [23]. Additionally, the majority of users who rely on the lock icon remain unaware of its identity feature [23,24,26,39] and do not reliably understand the concept of certificates [22, 23]. Indicators for newer technologies such as EV-SSL have also been shown to be ineffective to convey better security to the user as compared to a simple SSL certificate [18, 28].

TECHNIQUES FOR BETTER INDICATORS: Several techniques have been proposed to design better security indicators to prevent potential attacks such as phishing and web spoofing. Researchers have proposed better warnings [37], more effective interface dialogues [18], browser plugins [20], trusted path from the browser to the human user [40] and mandatory security indicators [27] to help users make correct security decisions. Other proposed security mechanisms include disabling JavaScript in the user browser and forcing persistent visibility of the browser's location line [25]. Dynamic Security Skins [22] allow a remote web server to prove its identity in a way that is easy for a human user to verify and hard for an attacker to spoof. Finally, efforts have been taken [6, 9, 10, 12, 16] to standardize security indicators and thus minimize confusion across browsers.

MOBILE BROWSER INDICATORS: Almost all the efforts in the area of security indicators in browsers have been focused on desktop browsers. The increasing user base of mobile web browsers and mobile e-commerce has made mobile browsers attractive targets for attacks [5, 8, 11, 21, 31–33]. In light of these developments and considering how the mobile browser user interface differs from desktops, it is important to analyze and understand the security indicators used in mobile browsers. Although the W3C [10] guidelines consider mobile browsers in their definitions, a large scale evaluation of the state-of-the-art security indicators in mobile browsers has not been carried out.

8 Concluding Remarks

Modern mobile browsers enable a range of sensitive operations over SSL/TLS connections. Although these browsers aim for equivalent functionality to traditional desktops, their smaller screen size has resulted in significant changes to the presentation and availability of SSL indicators. This paper presents the first large scale, cross-sectional measurement of this class of applications and compares the security indicators used in the overwhelming majority of mobile browsers to their traditional desktop counterparts. Our results are threefold: that mobile browsers implement only a subset of the recommended indicators from the desktop world thus eliminating the opportunity for even expert users to avoid attacks such indicators might signal, that the subset chosen across each browser are inconsistent and that the newest indicator (EV-SSL) is virtually unseen. Our measurements lead us to the conclusion that current security indicators force our community to either accept a false sense of security or to argue for the complete implementation of indicators (that are likely to be ignored) at the potential cost of overwhelming content. Having presented our empirical evidence of these problems, we argue for high profile exposure of our results within the security community. Moreover, we offer a straw-man proposal to motivate a call-to-arms to address mobile browser security interface challenges, and better allow informed decisions based on knowledge of the state-of-the-art.

The importance of mobile browsers is only increasing. In particular, a growing number of the most popular mobile apps (e.g., Facebook, ESPN, BankofAmerica) are simply wrappers for the mobile browser via APIs such as WebViews. The advantage to this approach is that it allows for a consistent user experience across virtually all platforms while drastically reducing the engineering effort to achieve this feat. In spite of being reliant on the browser, such apps provide users with *none* of the traditional indicators of security offered by the browsers. Specifically, these APIs present all of the content of the browser without any of the chrome, meaning that all primary and secondary security interfaces are absent. If such indicators are critical to providing security to browser users, the community must then confront the question directly and argue decisively for their use. If they are not, the community should instead argue to remove them entirely to avoid unnecessary use of precious real-estate, user confustion and a false sense of security. Regardless of which argument is made, the community must face these questions head-on.

Acknowledgments

This work was supported in part by the US National Science Foundation (CNS-0916047, CAREER CNS-0952959). Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation. We would like to thank N. Asokan, Jon Callas, Saurabh Chakradeo, Chaz Lever, Glenn Wurster and Mary Ellen Zurko for their feedback and comments.

References

[1] GoDaddy SSL certificate. http://www.godaddy.com/Compare/gdcompare_ssl.aspx?isc=sslqgo016b.

[2] VeriSign certificate. https://www.verisign.com/ssl/buy-ssl-certificates/index.html?sl=t72010166130000002\&gclid=CIKMyY2GuKgCFYg32godV2_8Bw.

[3] Key words for use in RFCs to Indicate Requirement Levels. http://www.ietf.org/rfc/rfc2119.txt, March 1997.

[4] The Transport Layer Security (TLS) Protocol Version 1.1. http://www.ietf.org/rfc/rfc4346.txt, April 2006.

[5] Overflow clickjacking. http://research.zscaler.com/2008/11/clickjacking-iphone-style.html, November 2008.

[6] Guidelines for the Processing of EV Certificates, version 1.0. http://www.cabforum.org/Guidelines_for_the_processing_of_EV_certificatesv1_0.pdf, January 2009.

[7] SSLstrip, presented at Black Hat DC. http://www.thoughtcrime.org/software/sslstrip/, 2009.

[8] Android Browser Exploit. http://threatpost.com/en_us/blogs/researcher-publishes-android-browser-exploit-110810, 2010.

[9] Guidelines For The Issuance And Management Of Extended Validation Certificates, version 1.3. http://www.cabforum.org/Guidelines_v1_3.pdf, November 20 2010.

[10] W3C: Web Security Context: User Interface Guidelines. http://www.w3.org/TR/wsc-ui/, August 2010.

[11] Web-based Android attack. http://www.infoworld.com/d/security-central/security-researcher-releases-web-based-android-attack-317?source=rss_security_central/, November 2010.

[12] Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates, version 1.0. http://www.cabforum.org/Announcement-Baseline_Requirements.pdf, April 11 2011.

[13] Comodo compromise. http://www.csoonline.com/article/678777/comodo-compromise-expands-hacker-talks, April 1 2011.

[14] DigiNotar CA compromise. http://community.websense.com/blogs/securitylabs/archive/2011/08/30/diginotar-ca-compromise.aspx, August 30 2011.

[15] Mobile Browser Market Share. http://gs.statcounter.com/#mobile_browser-ww-monthly-201011-201111, November 2011.

[16] The CA/Browser forum. http://www.cabforum.org/, April 11 2011.

[17] Android os market share by version. http://developer.android.com/resources/dashboard/platform-versions.html, February 2012.

[18] R. Biddle, P. Van Oorschot, A. Patrick, J. Sobey, and T. Whalen. Browser interfaces and extended validation SSL certificates: an empirical study. In *Proceedings of the ACM workshop on Cloud computing security*, 2009.

[19] M. Boodaei. Mobile users three times more vulnerable to phishing attacks. http://www.trusteer.com/blog/mobile-users-three-times-more-vulnerable-phishing-attacks, 2011.

[20] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. Mitchell. Client-side defense against web-based identity theft. In *Proc. NDSS*, 2004.

[21] C. Davies. iPhone Os Safari Vulnerable To DoS Attacks. http://www.iphonebuzz.com/iphone-safari-dos-bug-discovered-162212.php, April 16 2008.

[22] R. Dhamija and J. Tygar. The battle against phishing: Dynamic security skins. In *Proceedings of the symposium on Usable privacy and security*, 2005.

[23] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, 2006.

[24] J. Downs, M. Holbrook, and L. Cranor. Decision strategies and susceptibility to phishing. In *Proceedings of the Second Symposium on Usable Privacy and Security*, 2006.

[25] E. W. Felten, D. Balfanz, D. Dean, and D. S. Wallach. Intrusion Detection Prevention Web Spoofing: An Internet Con Game. In *20th National Information Systems Security Conference*, 1997.

[26] B. Friedman, D. Hurley, D. Howe, E. Felten, and H. Nissenbaum. Users' conceptions of web security: a comparative study. In *CHI extended abstracts on Human factors in computing systems*, 2002.

[27] A. Herzberg and A. Jbara. Security and identification indicators for browsers against spoofing and phishing attacks. *ACM Transactions on Internet Technology*, 2008.

[28] C. Jackson, D. Simon, and D. Tan. An evaluation of extended validation and picture-in-picture phishing attacks. *Financial Cryptography and Data*, 2007.

[29] B. Livshits and D. Molnar. Empowering Browser Security for Mobile Devices Using Smart CDNs. In *Proceedings of the Workshop on Web 2.0 Security and Privacy (W2SP)*, 2010.

[30] M. Marlinspike. More Tricks For Defeating SSL In Practice. http://www.blackhat.com/presentations/bh-usa-09/MARLINSPIKE/BHUSA09-Marlinspike-DefeatSSL-SLIDES.pdf, 2009.

[31] Y. Niu, F. Hsu, and H. Chen. iPhish: Phishing Vulnerabilities on Consumer Electronics. In *Usability, Psychology, and Security*, 2008.

[32] A. Porter Felt and D. Wagner. Phishing on mobile devices. In *Web 2.0 Security and Privay*, 2011.

[33] J. Resig. iPhone overflow clickjacking. http://ejohn.org/blog/clickjacking-iphone-attack/, November 2008.

[34] S. Schechter, R. Dhamija, A. Ozment, and I. Fischer. The Emperor's New Security Indicators. *IEEE Symposium on Security and Privacy*, 2007.

[35] J. Sobey, R. Biddle, P. van Oorschot, and A. Patrick. Exploring user reactions to new browser cues for extended validation certificates. *European Symposium on Research in Computer Security (ESORICS)*, 2008.

[36] D. Stebila. Reinforcing bad behaviour: the misuse of security indicators on popular websites. In *Proceedings of the 22nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*, 2010.

[37] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. 18th USENIX Security Symposium Crying Wolf: An Empirical Study of SSL Warning Effectiveness. *Work*, 2009.

[38] N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux. The inconvenient truth about web certificates. In *The Workshop on Economics of Information Security (WEIS)*, 2011.

[39] T. Whalen and K. Inkpen. Gathering evidence: use of visual security cues in web browsers. In *Proceedings of Graphics Interface*, 2005.

[40] Z. E. Ye, S. Smith, and D. Anthony. Trusted paths for browsers. *ACM Transactions on Information and System Security (TISSEC)*, May 2005.

A Appendix

A.1 Definitions

We define the terminology used in the W3C guidelines referenced within this paper.

User interface elements: User interface elements in browsers are divided in two categories [10]:

- *Primary User Interface:* the portions of a user interface that are available to users without being solicited by a user interaction. The primary user interface elements related to security traditionally include the padlock icon, the address bar, the `https` URL prefix, the favicon, and the site-identity button or URL coloring to signify the presence of EV-SSL and SSL certificates [9].

- *Secondary User Interface:* the portions of a user interface that are available to the user after they are solicited by a specific user interaction. The secondary user interface elements related to security include the security properties dialog, domain name, owner information, verifier information, information on why a certificate is trusted, validity period of manually accepted certificates (self-signed) and cipher details of an SSL connection.

Trust anchor: A trust anchor represents an authoritative entity represented by a public key and associated data. The public key is used to verify digital signatures and the associated data is used to constrain the types of information for which the trust anchor is authoritative. Relying parties (web browsers) use trust anchors to determine if digitally signed information objects are valid by verifying digital signatures using the trust anchor's public key and by enforcing the constraints expressed in the associated certificate data. Our interpretation is that a trust anchor refers to a certificate authority (CA).

Root: A root is a trust anchor that is any certificate authority (CA).

Trusted root: A trusted root is a CA whose public key is a priori trusted by the browser and may certify other keys.

Certificates: Public key certificates are widely used to provide keying material and convey a website's identity information to the user. The W3C defines four types of certificates. We provide our interpretation for the definitions of certificate types in the W3C document where they are ambiguous. For additional information regarding the commercial practice of issuing and managing SSL certificates, please refer to the requirements defined by the CA/Browser forum [12].

- *Validated certificate:* This is a public key certificate that has been verified by chaining up to a trusted root. Our interpretation is that a standard SSL certificate signed by a CA trusted by a browser refers to a validated certificate.

- *Augmented assurance certificate:* The certificate chain for such a certificate MUST be validated up to a trusted root that is recognized as augmented assurance qualified by the user agent (user's browser). We interpret an EV-SSL certificate as an augmented assurance certificate that is validated by the browser.

- *Self-signed certificate and untrusted root certificate:* A self-signed certificate is a certificate that is signed by its own creator and is not a priori trusted by a browser. Our interpretation of an untrusted root certificate is that it refers to a certificate holding the public key of a CA, that is signed by a CA not a priori trusted by the user's browser.

- *Interactively accepted trust anchors or certificates:* This refers to either a CA or a website's public key that is accepted by a user and thereby used as a trust anchor by the browser. Whether the trust anchor is accepted just for the present transaction or for the present and the future transactions depends on the options presented to the user by the browser and then the option chosen by the user.

  When a browser receives a website certificate, the public key therein (and the certificate) is untrusted

unless either the certificate was previously interactively accepted (for future sessions), or trust can be derived in it transitively, through a trust chain starting from a trust anchor (i.e., a CA key already trusted by the browser).

Pinning: Pinning associates one or more certificates with a specific website. The certificate provided by the website can either be self-signed or one issued by an untrusted root. Once a user interactively accepts such a certificate for the first time, the browser pins the certificate to the website. After pinning, the browser warns users only when the same website presents a different certificate. No warning messages are shown by the browser if a site shows a certificate consistent with previously pinned certificates for that site.

Identity Signal: An identity signal on a TLS-secured webpage includes information about the owner of the webpage and the certificate issuer's organization. A webpage's certificate provides its owner information and the issuer's (e.g., Certificate Authority) organization.

Strong TLS: An `http` transaction is strongly TLS-protected if it is TLS-protected, an `https` URL was used, strong TLS algorithms were negotiated for both confidentiality and integrity protection, and at least one of the following conditions is true: the server used a validated certificate that matches the dereferenced URI; the server used a self-signed certificate that was pinned to the destination; the server used a certificate chain leading to an untrusted root certificate that was pinned to the destination.

A strong TLS algorithm implies that no version of the TLS protocol that suffers known security flaws has been negotiated. Therefore, versions of SSL prior to SSLv3 MUST NOT be considered strong. Additionally, a strong TLS algorithm must also select a cipher suite for which key and algorithm strengths correspond to industry practice. More information on strong and weak TLS algorithms can be found in the W3C document [10] and RFC 4346 [4].

Weak TLS: An `http` transaction is weakly TLS-protected if it is TLS-protected, but strong TLS protection could not be achieved for one of the following reasons: TLS handshake used an anonymous key exchange algorithm, such as DH_anon; the cryptographic algorithms negotiated are not considered strong, such as TLS_KRB5_EXPORT_WITH_DES_CBC_40_SHA; certificates were used that are neither validated certificates nor self-signed certificates pinned to the destination.

Error messages: The W3C document defines common error interaction requirements and practices to signal two classes of errors ordered by increasing severity: warning/caution messages and danger messages.

Warning/caution messages are intended for situations when the system has reason to believe that the user may be at risk based on the current security context information, however a determination cannot positively be made. Danger Messages are intended for situations when there is a positively identified danger to the user (i.e., not merely a risk).

A.2 Additional Results

We note additional observations on the positive and negative characteristics shown by the mobile and tablet browsers. Note that these findings are not directly related to the guidelines studies in Section 3.

A.2.1 The Good

The W3C document defines two guidelines that MUST hold when strong TLS algorithms are negotiated between a client and a server:

1. No version of the TLS protocol that suffers known security flaws has been negotiated. At the point of writing of this document, versions of SSL prior to SSLv3 MUST NOT be considered strong.

2. A cipher suite has been selected for which key and algorithm strengths correspond to industry practice. The "export" cipher suites explicitly prohibited in appendix A.5 of TLSv11 [4] (RFC 4346) MUST NOT be considered strong.

To verify the compliance with these guidelines we conducted two experiments.

SSLv2: We browsed to a website supporting only SSLv2 from each of the candidate browsers. We found that all the mobile, tablet and desktop browsers comply with the first guideline and do not support SSLv2.

Null cipher: The null cipher is one of the prohibited ciphers in RFC 4346 and one of the most dangerous ciphers because it represents the lack of an encrypted communication channel. To test browser compliance with the second guideline for strong TLS algorithms, we built a website that supports only the null cipher. We observed that none of the mobile, tablet or desktop candidate browsers support the null cipher.[3]

Discontinuing support for SSLv2 and the null cipher automatically reduces the probability of cipher downgrade attacks on the candidate browsers.

A.2.2 The Bad

[3]We did not test for the support to all the prohibited ciphers (as given in TLSv11 [4]) by the candidate browsers.

Mobile and Tablet Browsers (See Table 1 for versions)	SSLv2 not supported?	Null cipher not supported?	Weak cipher prohibited? (DES-CBC-SHA)
Android	•	•	×
Blackberry Mango	•	•	×
Blackberry Webkit	•	•	×
Chrome Beta	•	•	•
Firefox Mobile	•	•	•
iPhone Safari	•	•	×
Nokia Browser	•	•	×
Opera Mini	•	•	•
Opera Mobile	•	•	•
Windows IE Mobile	•	•	•
Safari on iPad 2	•	•	×
Android on Galaxy	•	•	•

Table 9: Results of the support for SSLv2, the null cipher and DES-CBC-SHA (weak cipher). The symbol notation is as defined in Table 2.

A browser supporting a weak cipher can enable a network attacker to break the encrypted messages offline. The SSLv3 cipher-suite consists of certain weak ciphers, although they are stronger than the SSLv2 ciphers and the null cipher. We verified the support of the DES-CBC-SHA weak cipher. We observed that six (Android Mobile, Blackberry Mango and Webkit, iPhone and iPad2 Safari and Nokia) out of the eleven mobile and tablet browsers support the weak cipher. The other mobile and tablet browsers display error messages conveying the absence of a common encryption protocol with the server. It is interesting to note that the the Safari browser in its mobile, tablet and even desktop versions supports this weak cipher. However, the Android tablet browser does not support this cipher, unlike its mobile version. Since most mobile and tablet browsers do not allow users to see the cipher used on a TLS connection, they can not determine that a weak cipher is being used. No desktop browser other than Safari supports this cipher.