ABSTRACT

One of the latest attacks on secure socket layer (SSL), called the SSLstripping attack, was reported at the Blackhat conference in 2009. As a type of man-in-the-middle (MITM) attack, it has the potential to affect tens of millions of users of popular online social networking and financial websites protected by SSL. Interestingly, the attack exploits users’ browsing habits, rather than a technical flaw in the protocol, to defeat the SSL security. In this paper we present a novel approach to addressing this attack by using visually augmented security. Specifically, motivated by typical traffic lights, we introduce a set of visual cues aimed at thwarting the attack. The visual cues, called security status light (SSLight), can be used to help users make better, more informed decisions when their sensitive information need to be submitted to the websites. A user study was conducted to investigate the effectiveness of our scheme, and its results show that our approach is more promising than the traditional pop-up method adopted by major web browsers.

1. INTRODUCTION

There has been growing interest in usable security over the last decade, and numerous studies have been conducted to explore the usability issue of various computer security mechanisms [8, 19, 21, 22]. However, bridging the gap between security and usability is a challenging task. At one extreme, we can easily find security technologies, for example, one-time pads (OTPs), that render many sophisticated attacks obsolete but are not widely adopted due to their usability issues. At the other extreme, we can find security solutions, for instance, low entropy passwords, that are very usable but fail to withstand non-trivial attacks.

One of the common mistakes made when designing security systems is that security is assumed to be both important and engaging for users. However, this is clearly not a correct assumption since security is almost never the users’ primary goal [22]. The security ramification of this disengagement is the constant surfacing of new attacks that exploit not only technological flaws, but also usability flaws. One such example is the SSLstripping attack introduced at the Blackhat conference in 2009 [11]. It attacks secure socket layer (SSL), which is the most widely used security mechanism that enables secure communication establishment between two parties over the Internet. As a type of man-in-the-middle (MITM) attack, the attack could affect tens of millions of online users of popular SSL-protected websites such as Facebook.com. More importantly, the attack exploits users’ browsing habits, rather than a technical flaw in the protocol, to effectively defeat the SSL security.

The underlying cause of this type of semantic attack, which targets the way we, as humans, assign meaning to content [14], is a semantic barrier between the user’s mental model and the system’s actual processing model [21]. Since web interactions between a user and a remote web server are usually made through the use of a web browser, the user derives her mental model of the interactions from the presentation of the interactions - the way they appear on the web browser screen. On the other hand, the system’s processing model is derived from the messages exchanged between the web browser and the web server, without a web browser being aware of the user’s intention. Hence, the user, who usually does not look at the hidden HTML codes, is not able to compare her mental model (“I use a secure channel to submit my login credential”) with the system’s processing model (“A secure channel is not used due to an attack”) under the SSLstripping attack. The semantic barrier further results in the lack of trust between the user and her agent.

In this paper we present a novel approach to addressing this type of semantic attack, especially, the SSLstripping attack, by using visually augmented security. Specifically, motivated by the red-yellow-green colored traffic light metaphor, we introduce a set of visual cues aimed at thwarting the attack. The visual cues, called security status light (SSLight), can be used to boost the user’s trust against her browser by trying to overcome the semantic barrier between the user’s mental model and the system’s processing model. Hence, users can make better, more informed decisions when their sensitive information needs to be submitted to a remote website. An experimentation involving a user study was designed and conducted to investigate the effectiveness of our scheme, and its result shows our approach is more effective than the traditional pop-up method adopted by major web browsers.

This paper is organized as follows: Section 2 discusses the SSLstripping attack and its countermeasures in detail. Section 3 presents our approach to tackle the attack, with an
introduction of SSLight. Section 4 discusses the methodology and design of a user study developed to test the effectiveness of our approach. Section 5 and 6 present the results of the user study. Section 7 concludes this paper with our future research direction.

2. BACKGROUND AND RELATED WORK

Since SSL was first introduced by Netscape in 1994, it has become a de facto standard for establishing a secure connection between two parties over the Internet. It also laid the foundation for developing a new standard called transport layer security (TLS) [4, 5]. Nowadays, many websites rely on HTTPS, HTTP served over a SSL/TLS channel, to provide users a secure transaction over the Internet. It is interesting, however, to see that nearly 45% of the most popular websites still do not use HTTPS, not even for login purposes, as shown in a recent study [16].

2.1 SSLstripping Attack

The man in the middle (MITM) attack has been a constant threat in web security. A MITM attack is achieved when a malicious user positions himself between two other users that intend to engage in a legitimate communication. Effectively, in a MITM attack, the attacker impersonates each of the victims to the other. A variation of MITM attack called the SSLstripping attack was proposed at the Blackhat conference in 2009 [11]. The attack takes advantage of the simple concept and observation that most users do not explicitly type the safe address of a web page (https), but rather rely either on the browser or the target page to redirect them to a secure location. This opens the opportunity to strip users’ sessions of its security, while giving the user the illusion of privacy. More specifically, the attack works as follows:

- The attacker is on the same local network as the victim
- The attacker uses the ARP protocol to convince the victim that the attacker’s machine is the victim’s gateway
- The victim, unknowingly, is now sending all her HTTP requests to the attacker
- The attacker will look at all HTTP requests and do either of the following:
  - Case 1: If the response is a redirection to an HTTPS address, the attacker will establish the secure connection to the address and provide the decrypted content to the victim, also stripping ‘s’ from “HTTPS” in any forms and links
  - Case 2: If the response is a mixed content page, which is an HTTP page with forms with “HTTPS” actions, the attacker will simply strip ‘s’ from “HTTPS”
- The victim logs in to a service
- The attacker will receive the victim’s login request and:
  - Register the victim’s credentials
  - Login in place of the victim with the web server
  - Provide the victim with the received responses, after decryption
- The victim, unaware of the attack, experiences the same as she would experience if the connection was secure. Her browser issues no warnings, since no SSL certificates are being used on the victim’s machine.

As shown by Marlinspike at the conference and verified later at our lab, this attack is really easy to launch and extremely effective: it would be detected only by very careful and technically savvy users by either noticing the insecure connection on a normally secure website (Case 1) or by looking at the rendered web page’s source code and verifying that the login form does not have a secure address as its target (Case 2). Figure 1 illustrates this attack by using Facebook.com.

2.2 SSLstripping Countermeasures

Most of the current web browsers have little or no support for security indicators when users submit their sensitive information over an insecure communication channel. Hence, it is hard for users to have any suspicion that something should be wrong when they are under the SSLstripping attack. Both Mozilla Firefox and Internet Explorer support...
a warning dialog when users submit their data over an insecure channel, as shown in Figure 2. However, this is an opt-in feature which users need to explicitly choose to see the warning. Other browsers, such as Google Chrome, do not have even the opt-in warning dialog. However, a warning feature could be supported by installing browser extensions in Google Chrome.

Jackson and Barth suggested a solution where the user’s web browser would force a secure connection with websites that have previously deployed a special cookie [9]. This solution will not be able to avoid the SSLstripping attack in general, since most websites do not have the SSL-only requirement. Nikiforakis et al. proposed an approach to leveraging a client side HTTP proxy, which, relying on browser history, would compare the current request with previous requests made to the same website in [12]. Although this is an interesting approach, it fails to address several issues. First of all, it requires a browser history. This automatically excludes all users that do not save browsing history. Second, it assumes that the attack is never a zero day attack. This excludes all users that were already attacked. The assumption that the attack will only occur after a significant browser history is established, seems to be a very weak assumption.

2.3 HTTPS Visual Security Cues

Whalen and Inkpen conducted a study on the effectiveness of different security indicators in [18]. Their study used an eye-tracking device to match users’ claims to their actual actions, more specifically where they focused their attention. The study had several interesting results. First of all, on top of the previously identified security indicators such as a padlock and https in an address bar, more security indicators were identified, specially concerning a web site reputation, either by general user rating or by the level of usage among a user’s circle of friends. Furthermore, the study was run in two phases, the first where users were not prompted to pay special attention to security aspects, while in the second they were. They showed that the resulting differences between the two stages of the study were very significant: users would normally not pay attention to the security indicators. Overall, the lock displayed on a browser in the event of a secure connection being established was the preferred security indicator. However, it is important to notice that many browsers allow a page to display a small icon on the address bar, which can be made to look like a lock regardless of a secure connection being established or not. Additionally, it was found that some users were confused with other small icons for the lock icon.

In their study, Schechter et al. found out that removing the SSL security indicators would not deter users from logging in to different websites with their credentials [13]. But this phenomenon could be explained partially by the existence of different security indicators, not directly related to SSL, as identified by Whalen and Inkpen [18]. On the other hand, the effect of introducing extended certificates, aiming to provide even more information about service providers to users, was studied in [10, 15]. However, it was found that users also ignored the information on these certificates.

Usable security issues were not limited to the SSL usage for secure web browsing. There have been several studies on the usability of PGP encryption and effectiveness of anti-phishing systems in [6, 20, 21], and they showed that the design of a usable security system is a hard problem. The general consensus is that security indicators that rely on the user to make a correct decision tend to be ineffective in [2, 3, 7]. This is also supported by Sunshine et al., who concluded that warnings should be avoided when possible and decisions should be made for the user in an automated, under the hood fashion [17]. On the other hand, Whitten and Tygar showed that a careful design can lead to much more usable security in [20]. Brustoloni and Villamarin-Salomon presented promising results by trying to prevent users from getting habituated to existing warning dialogs and introducing morphic warning messages in [1].

3. OUR APPROACH

Our approach to addressing the SSLstripping attack is based on the use of visual security cues as a way to help overcome the semantic barrier between the user’s mental model and the system’s processing model. Specifically, the visual security cues will be displayed in login form fields so that users will be able to detect the attack based on them. Because of a preference of usability over security, visual security cues are often disregarded by users. Therefore, our challenge is to design and build a security solution that gives more value to users than the value of the effort required to use it. Unlike regular software systems, which users use because there is some kind of return for them, security is most of the times an add-on, and if the effort to include security is much higher than the perceived risk the user is exposed to, users will just ignore it. Hence it is of utmost importance to design a more easy-to-use and more intuitive, but less disruptive visual cue solution.

The design of our visual cues was motivated by the popular traffic light metaphor, which is simple to understand and familiar to most users. This is primary reason behind our adoption of the traffic light metaphor, and we strongly believe that it can be used to help simplify the decision making management for both lay and technically savvy users when they are about to submit their sensitive login credential. As we discussed previously, the attack is so effective that most users are not likely to be able to detect it. Therefore, we strongly believe that each security indicator simply represented by a single traffic light, red, yellow, or green and displayed inside of login form fields will better assist them to understand the current security situation that they are faced with and to make better, more informed decisions when they need to submit their sensitive information to a remote website. Motivated by this, our solution called security status

![Figure 2: The classic Pop Up Window warning present in Mozilla Firefox](image-url)
light (SSLight) adopts the three color design, as shown in Figure 3.

In following sections we first discuss an algorithm to display SSLight so that it can be used in judging whether login credentials are to be submitted securely or not. Next, we implement a browser extension that executes the algorithm and presents the result to the user.

3.1 SSLight

Before the user can be given some information regarding the website where she is about to log in, we need to be able to evaluate the security of this website. We developed an algorithm that will look at the web page source code and output an evaluation based on the comparison of the web page’s address and the login form’s action data, which has an URL where credentials will be submitted.

The first step is to identify if the page loaded by the web browser is already being accessed over SSL. If this is the case, we just need to verify that the action on the login form belongs to the same domain that is already secure, a situation that is true if the action is a relative path. This means the form will submit the login request to an address in the scope of the current secure connection. Hence, the SSL stripping is not possible and we return a positive evaluation (Green light). If the current page is not on a secure connection, we will assess the protocol being used in the form action. If we find that the form action URL is an insecure address, we immediately return a negative evaluation (Red light), which means it is unsafe to submit the login request.

In the scenario where the current page is not secure, but the form action is under the HTTPS protocol, we proceed with another round of analysis. Further analysis will first assess the certificate of the secure location referenced in the form action. If this is a self-signed or expired certificate, we will return a negative evaluation. Next, if the certificate proves not to be invalid, we compare the domain in the form action with the domain of the loaded page. If these two domains match, we return a positive evaluation, otherwise, we check whether the domain of the login form action is in a list of trusted login entities. If we cannot white-list the URL, we will issue an uncertain assessment (Yellow light) and delegate to the user, showing them the domain where they will be submitting their form. In an attack situation, a warning would appear in the webpage that previously did not raise any issues.

Our algorithm is outlined in pseudo code in the following listing:

3.2 Browser Extension Implementation

SSLight was implemented as a browser extension on Google Chrome web browser using Javascript, HTML and CSS. This implementation should be able to be ported with the necessary adaptations to every other browser that allows extension development. The extension executes only after the page has been loaded, being the last layer of code to be executed on the rendered page. The algorithm we designed is run and after a conclusion is made, the result is presented to the user.

We chose to implement two different visual cue solutions. The first is SSLight present at all times in the login form fields in the loaded website. The SSLight will spin like the images in slot machines when the page is loaded and assessed, and will stop spinning and display an appropriate color when the assessment is made. The green light indicates that the form is secure, being submitted through a secure channel. The red light means the form is insecure and the yellow light represents a very small number of cases where we cannot make a definite assertion and delegate it to the user. This visual cue can be seen in Figure 4. Note that a tool tip warning message associated with each of the colored lights is not displayed here. However, they are used in our user study to deal with color-blinded people.

The other visual cue solution is to simply change the background color of the input boxes, either blinking or not, to a red color that alerts the user to danger. In the second solution, shown in Figure 5, nothing will be presented to the user if the assessment returns an output that signals a

```plaintext
EVALUATE-FORM()

1. if HTTPS-INITIAL()  
   2. return green-signal
   3. else
   4. if formAction ≠ https
      5. return red-signal
   6. else
   7. FURTHER-ANALYSIS()

FURTHER-ANALYSIS()

1. if ~VERIFY-SSL-CERTIFICATE()  
   2. return red-signal
   3. else
   4. if form.act.loc.hostname ≠ doc.loc.hostname
      5. if WHITE-LIST(form.act.loc.hostname)
         6. return green-signal
      7. else
         8. return yellow-signal
      9. else
         10. return green-signal
```
To be shown. By non-disruptive we mean that the warning would be displayed. It was what the users decided to do at this point that we were interested in.

Since we did not want to expose the users to any real danger, i.e. setting up a real SSLstripping attack where user credentials are sent unencrypted over the network and could be read by anyone sniffing it, we prevented an actual login from occurring on the second machine and redirected the users to the exit survey instead. The users who chose not to continue after noticing the security warning, were asked to login with no credential information, and therefore also redirected to our exit survey. Before the beginning of the study we defined a set of hypotheses that we wanted to have tested. The hypotheses we developed are:

1. (General awareness of secure form submission) Users will know the difference between encrypted and unencrypted credential submission when they log in to a website.

2. (Effectiveness of SSLstripping, given the awareness) Users will not notice any difference when they submit their credential to log in to a website under a successful SSLstripping attack.

3. (Unhelpfulness of an existing method) Users will submit their credential, even when warned with the classic pop-up alert message for unencrypted data submission.

4. (Helpfulness of visual cue-based methods) Users will not submit their credential, when warned with new visual cue-based approaches.

5. (Effectiveness of different visual cue methods) There is no difference in the submission ratio between blinking background and SSLight approaches.

The first hypothesis assumes that the average user is aware of the difference between secure and insecure form submission. In other words, the average user knows that a secure form submission encrypts the form data before sending it over the network. Given this knowledge, the second hypothesis tests whether users are able to detect the SSLstripping attack, meaning that users will not notice when a normally secure website submits their credentials unencrypted, the consequence of being under the SSLstripping attack. We hypothesize that the average user will not be able to detect the attack. The third hypothesis tests the assumptions that the current warning method, a pop-up window with

4. USER STUDY

We conducted a user study not only to test the effectiveness of our solution, but also to compare it against an existing solution, namely the classic pop-up window in Firefox. We also wanted to study how all of these solutions compare against the absence of any visual security cue. Finally, we wanted to test all of the warnings in the presence of the SSLstripping attack.

To test different scenarios that will be elaborated later, we devised four different user groups, each exposed to the attack and a different kind of warning or no warning at all. We wanted to investigate how effective our proposed solution is against an SSLstripping attack.

1This user study was approved by our university’s IRB and conducted for one week at our university campus.
security warning message, will be ineffective to prevent the attack and users will still submit their credentials, despite the warning that the current page is submitting credentials unencrypted and despite the fact that in the previous login activity there was no warning. The last two hypotheses are related to our proposed approach. We hypothesize that our approach will be more effective than the traditional approach, and also that our two different visual cues will have identical effects.

4.1 Experimental Design

We defined four separate user groups, each of which is exposed to a different warning scenario:

- Group 1: Exposed to the attack with no warning
- Group 2: Exposed to the attack with the standard pop-up warning dialog
- Group 3: Exposed to the attack with the SSLight warning in the login form fields
- Group 4: Exposed to the attack with the blinking background in the login form fields

The users were asked to act as if they were using their own machines, in that all decisions they made should be the same as if they were being made on their own private computers. Security was never explicitly mentioned. Even though we initially thought the “make all decisions as if this was your machine” could bring too much focus into security issues, the results showed us that this was not the case. To avoid making the user aware of our real purpose, the experiment script was carefully designed so that the users thought they were being tested on the features of the website being accessed and not on the security of the login action itself. Users were assigned randomly to one of the user groups and given instructions. The instruction sets were identical for all the users except the username with which they were to login to the test machines. Each of the four groups was assigned a different username corresponding to a different setup on the two test machines.

4.2 Sample

In the design of our experiment, we performed a power analysis to determine the minimum sample size that we would require to test our hypotheses. We chose an error of 0.05 and a power of 0.8, common among such experiments, and determined a minimum sample size of 19 subjects per study group, adding up to a total of 76 subjects across the four user groups.

In our study, volunteers were elicited by using fliers posted around our University campus and offering an iPod Touch as a prize to be drawn among the participants. A total of 106 individuals replied to our request, from which 5 did not perform the study and one submitted an empty exit survey form, which was discarded. A total of 100 valid submissions were obtained, 25 individuals per group, therefore above our minimum sample size and enough to test our hypotheses. The subjects who participated in our study mostly came from our university. The demographic details will be presented in the beginning of Section 5.

2Note that the users used their own username and password to login to facebook however.

Figure 6: Distribution of computer usage expertise among study subjects.

4.3 Exit Survey

The last step asked to the participants of our study was to answer a brief exit survey that included questions on the overall understanding of a secure form submission and its consequences, the presence of warnings in browsers and the user opinion on the different warning mechanisms. Even though each participant was exposed to only one scenario in the study, they were shown all the three warnings in the exit survey and asked to classify them in a 5 point Likert scale. The users were also asked to select their favorite warning method, out of the methods presented and an “other” option, that would allow users to write any other method of insecure form submission warning they would prefer over the presented possibilities. The users were also asked to quantify their concern with the security of their credential submission to a website. We used a nine-point Likert scale for this end. Finally, the users were asked to answer a few questions on their information, such as gender, education and computer experience level.

5. USER STUDY RESULTS

Our sample includes 100 individuals, whose computer expertise is on Figure 6 and education level is on Figure 7. Our subjects have a high education rate, with all having completed at least high school and the gross of participants being undergraduate college students. Relative to computer usage experience, the individuals in our sample are very sophisticated, with only 9 claiming they have basic skills, 39 and 41 claiming they have intermediate or advanced skills and 10 rating themselves as expert computer users.

5.1 Hypothesis 1: General Awareness of Secure Form Submission

This hypothesis was tested by asking all the users about their understanding of secure form submission in the exit survey. On the exit survey, we asked users to select all the statements from a list that applied to unencrypted form submissions. The summary of results is on Table 1. The statements are ordered in decreasing order on the number of users that selected that statement as applicable to an insecure form submission.

As it can be seen from the results, 89% of the subjects are aware that an insecure form submission exposes their
credentials to eavesdroppers on their local network, while only 66% are aware that credentials submitted insecurely are not encrypted by their browsers. While 42% of the subjects think an insecure form submission is caused by the network they are connected to, 32% believe an insecure form would be the cause of their private information being leaked by a service to third party entities, which is not dependent upon the submission security, but the service being logged in to. From these results, our first hypothesis is confirmed, and we can assume that the average user is aware of the dangers of submitting data over the network using forms over plain http.

5.2 Hypothesis 2: Effectiveness of SSLstripping attack

On our second hypothesis, we assumed that the SSLstripping attack would be highly effective. This means that the average user, being exposed to the attack, will not be able to detect it and will submit the form as if it was secure. To test this hypothesis, we had the first group of users exposed to the attack and not shown any warning. Facebook has the login form submitted to an HTTPS address by default, and after the login is performed, the user gets redirected to an HTTP address again. We postulate that the average user will not be aware of this situation, but there is still the possibility that a savvy user would be able to detect the attack by inspecting the source code.

Of the twenty-five users in the first user group, none of them detected the attack and all submitted the login form. This 100% effectiveness in the sample translates into an interval from 88.7% to 100% in the population, with 95% confidence. This interval confirms our hypothesis and allows us to assume that without any added security mechanisms, the SSLstripping attack will be highly effective against the average user.

5.3 Hypothesis 3: Unhelpfulness of Existing Method

The method for warning users of an insecure form submission and the dangers associated with it has traditionally been a pop-up window that appears after they entered their credentials in the form fields and clicked the submit button associated with the form. This method was present in Netscape Navigator and is currently an optional feature on Mozilla Firefox and Internet Explorer.

We hypothesized that the average user, when presented with this warning, would ignore it and proceed with the login operation. In our experiment the users were requested to login to Facebook.com on two different machines. On the first of these machines no warning was displayed, but on the second machine, the one under attack, Firefox would warn the users of the insecure form submission via the optional pop-up window. Table 2 shows the results we obtained, compared with the results obtained for the absence of any warning and the statistical analysis of difference between these two groups.

With only one user out of 25 refusing to continue when presented with the pop-up window, the statistical test reveals that there is no meaningful difference between having no warning and having the pop-up window.

5.4 Hypothesis 4: Helpfulness of Visual Cue Methods

The main goal of our user study was to investigate the effectiveness of the novel visual cue methods we proposed to notify the user of a threatening situation, namely the submission of a form containing sensitive credential information over an unencrypted communication channel. The previous two hypotheses were established as assumptions that we made during the design and development of our approach. While testing our solution, we also needed to establish if our assumptions were correct, which was verified by failing to reject the null hypotheses in Section 5.1 and Section 5.2. Having support for these two assumptions, we can now compare them to the results obtained with the two user groups exposed to our proposed warning methods. The summary of our results and of the statistical significance of the difference between the different user groups is shown in Table 3 and 4.

![Figure 7: Study subjects education level.](image)

### Table 1: Unencrypted form submission danger awareness

<table>
<thead>
<tr>
<th>Option</th>
<th>Positive Answer Count</th>
<th>95% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Your username and password can be seen by an attacker connected to your local network</td>
<td>89</td>
<td>0.811 - 0.944</td>
</tr>
<tr>
<td>Your browser does not encrypt your password</td>
<td>66</td>
<td>0.558 - 0.752</td>
</tr>
<tr>
<td>The encryption of your network is weak</td>
<td>42</td>
<td>0.322 - 0.523</td>
</tr>
<tr>
<td>The website to which you are submitting form will sell your information</td>
<td>32</td>
<td>0.23 - 0.421</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Fisher’s Exact P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Your username and password can be seen by an attacker connected to your local network</td>
<td>0.887</td>
</tr>
<tr>
<td>Your browser does not encrypt your password</td>
<td>0.950</td>
</tr>
<tr>
<td>The encryption of your network is weak</td>
<td>0.888</td>
</tr>
<tr>
<td>The website to which you are submitting form will sell your information</td>
<td>0.888</td>
</tr>
</tbody>
</table>

### Table 2: Comparison of having the Pop-up Window and having no warning

<table>
<thead>
<tr>
<th>Option</th>
<th>Submit</th>
<th>Not submit</th>
<th>Fisher’s Exact P</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Warning</td>
<td>25</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Pop-up Window</td>
<td>24</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

The table shows the comparison of having the Pop-up Window and having no warning.
We can see from Table 3 that 16 out of the 25 users in the SSLight user group submitted the form, while 9 refused to continue. This shows a significant difference to the proportion found in the pop-up window user group and confirms hypothesis 4 for our first approach. The second solution we proposed had only 8 out of the 25 users in the user group submitting the form. A total of 17 users refused to continue in the presence of this warning method. The statistical difference between this group, and the group exposed to the classic pop-up window is even more evident than that of our first solution. The results are summarized in Table 4.

Hypothesis 4 is therefore confirmed for both our solutions, meaning that they are indeed more effective than the traditional approach to warning a user of an insecure submission, as we failed to reject both the null hypotheses regarding the visual cues.

### 5.5 Hypothesis 5: Effectiveness of Different Visual Cue Methods

Our fifth and last hypothesis stated that both of our visual cue solutions would be equally effective, with no statistical difference between the ratio of users that ignored it and the ratio of users that refused to submit their credentials. The results, summarized in Table 5, show that there is indeed a statistical difference and therefore the null hypothesis is rejected. This means that the blinking background was statistically more effective than the SSLight solution. Failing to support hypothesis 5 indicates that our visual cue mechanisms are not equally effective, even though both are more effective than the classic approach.

### 6. OTHER RESULTS AND DISCUSSION

#### 6.1 Preferred Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Submit</th>
<th>Not submit</th>
<th>Fisher's Exact P</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSLight</td>
<td>16</td>
<td>9</td>
<td>0.011</td>
</tr>
<tr>
<td>Blinking</td>
<td>8</td>
<td>17</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Table 5: Comparison of having our SSLight Solution or our Blinking Background Solution

One of the questions included in our exit survey was relative to all the methods presented to the different groups. The question asked each individual in our user study to choose a favorite warning method out of the three methods used: pop-up window, SSLight and blinking background. A final option tagged “other” accompanied with an empty text field was also provided to allow users the freedom of not choosing one of the mentioned solutions, but still answer the question.

Figure 8 shows the compilation of the answers to this question. The chart shows not only the total number of votes each method received, but also how the different user groups voted across the different methods. It is interesting to notice that the only group that had no votes in the “other” option was the group that was exposed to no warning.

In addition to asking the users to select their preferred method, we asked them to rate each of the methods presented in the survey. The summary of the results is in Figure 9. The graph shows the average rating of each method along with the standard deviation. These results show that there is not a significant difference between the user rating of each method.

#### 6.2 Reaction to Warnings in the Past
Table 6: Distribution of users who have noticed insecure form submission warnings in the past and, among those who have, the fraction of those who opted out of them and those who did not.

<table>
<thead>
<tr>
<th>Noticed Warning in the Past</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deactivated Warning</td>
<td>84</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>47</td>
</tr>
</tbody>
</table>

Figure 10: Distribution of the concern about unencrypted login form submission.

Trying to understand the past user behavior in face of the classic pop-up window, we asked users if they had ever seen any warning about submitting form insecurely over the network. The results, summarized in Table 6, show that 84% of the users have in fact noticed an insecure form submission warning in the past. Of these, 43% say they opted out of the warnings and 56% claim they did not. The 1% missing is explained by a user who said he had seen a warning in the past but did not answer whether he opted out of it or not.

It is, however, important to note that at the time that this study was conducted, none of the major web browsers had a default insecure form submission warning.

6.3 User Concern

One of the first questions we asked in the exit survey was relative to the subject’s personal concern about insecure form submissions. On a scale of 1 to 9, being 1 not concerned and 9 very concerned, the average rating was 6.31, with a standard deviation of 2.246. It is then reasonable to assume the average user is somewhat concerned with the risks of submitting the credentials in plain text over the network. The distribution of the different ratings can be seen on Figure 10, where we can see an higher incidence on votes on the higher half than the lower half.

6.4 Alternate Methods

When we asked the users to select their favorite warning method, we also left room for the possibility that the users would rather not have any of the proposed methods, or even no warning method at all. We included an “other” option which also allowed them to write what would that other method be. The results collected from the users who chose “other” and provided further explanation include “a color-coded warning in the url bar”, “padlock-flashing red”, and “status message below the address bar”.

6.5 Potential Attacks

Considering that the SSLstripping attack is a MITM attack, where the attacker can change the content of the webpage being rendered by the browser, it is reasonable to assume that the attacker can change more than just the action attribute of a form to remove the ‘s’ from an address starting with “https”. We took special consideration to the scenario where the attacker assumes our extension is present and may try to circumvent it.

Our first concern is that an attacker could create an overlaying html component that could sit on top of the original form element, or just our warning, thus giving the user a false sense of trust by always displaying a positive warning. To avoid this attack scenario, our warning is rendered not on the form element itself, but on a separate layer placed on top of the form element. An alternate solution would involve having the shape of the warning, a circle in the case of SSLLight, to be defined by the user. This means that the attacker would have to correctly guess the shape chosen by the user for the warning in addition to the remainder of the attack. For the blinking background approach, we could allow the user to choose its own color scheme, forcing the attacker to have the burden of discovering this information.

We chose to implement an overlaying component as opposed to custom warnings, as it requires less effort from the user.

Another possible tweak to the attack that could be used to circumvent our approach is to include in the page a script that would change the action URL of the form only after our algorithm has finished running. This would mean that although the extension may have seen the original action address and it was deemed to be secure, the injected code could change it and the user would actually submit the form to an insecure location. To address this issue, our extension makes use of the existing Google Chrome extension API to verify the request after it has been made and double check the submission address. If at this point a risk is identified, the user will be warned and prompted for a confirmation.

It is noteworthy that both the scenarios above would require changes to the original SSLstripping attack that need to be tailored to individual web pages. The attack would no longer be as general and therefore not as powerful as the original SSLstripping attack.

7. CONCLUSION AND FUTURE WORK

In this paper we presented a novel approach to thwarting the SSLstripping attack by using visual security cues. Our empirical study clearly shows that the proposed solutions are more effective and efficient in preventing the SSLstripping attack than the classic pop-up window method. However, our approach is by no means complete. For our immediate future work, we will investigate how to improve the effectiveness of the SSLight solution. Specifically, the proposed SSLight is based only on color and text message. We will investigate how to add additional factors such as symbol or position to enhance its effectiveness. Another future work is related to our experimental design. Our study used a sample consisting mainly of higher education students, a demographic that does not represent the average user accurately. Gathering a more representative sample poses a bigger challenge that we are trying to address. There is also the fact that the results obtained by our approaches could stem directly from its novelty alone. To circumvent this, we
are working on another round of data collection that will require longer and more frequent interactions to exclude the novelty as a factor for the good results, thereby studying the effect of user habituation. Lastly, the usual rules that apply to a desktop browser are not applicable to a browser running on a smartphone, for example. We see the emergence of long solved security flaws in new devices, as is the case of the recent browser user interface spoofing in iPhone. These new exploitation opportunities come from the lack of consideration for security aspects in software development, specially in user interface design of mobile applications. We will investigate how our approach can be applied to this in the near future.

**ACKNOWLEDGEMENT**

This work was partially supported at the Secure Computing Laboratory at New Mexico Tech by the grant from the National Science Foundation (NSF-IIS-0916875).

8. REFERENCES


