Nabi - A Preventative Solution Against JavaScript and Scriptless Tabnabbing Attacks

Michael Baraboo, William Schattgen, Christopher To

4 May 2015

Abstract

1 Introduction

2 Background

2.1 TabShots

One approach to combating tabnabbing is a Google Chrome browser extension known as TabShots, which was created by Philippe De Ryck, Nick Nikiforakis, Lieven Desmet, and Wouter Joosen [1]. To detect whether a tab has changed or not while not in focus, TabShots uses visual comparison of the tab's appearance before and after the change in the user's focus. The extension records screenshots of the currently focused tab at regular intervals, using a simple Google Chrome API call and storing the screenshot as a data URL. The program also keeps record of the tab's favicon, or the icon displayed in the tab's url and title space. These two sets of images provide the basis for comparison when a tab regains focus.

When any tab regains focus, TabShots compares a new screenshot of the tab and it's current favicon to the most recent stored screenshot and favicon that was recorded before the tab lost focus. The small favicons are compared by source, and the screenshots are compared in a visually divided manner using the HTML5 canvas element. The screenshots are divided in a raster of fixed-size tiles, and each tile in the new screenshot is compared to the most recently stored screenshot of the tab. The tiles are sized as 10x10 pixels, which are deemed by the authors as a balance between performance and precision. If the tiles do not match exactly, that area is marked as changed. The HTML5 canvas element provides powerful image manipulation capabilities that allowed the rastering and comparison algorithms to be implemented.

Once the differences in the current tab from its previous version are calculated, an overlay is injected into the page that shows the differences in semi-transparent red. The overlay is both transparent in both visibility and mouse events, so no mouse and keyboard events will be affected it. The overlay is constantly checked for its presence by the extension whenever an element is removed, ensuring that a malicious page will not be able to remove the overlay without notifying the extension and user. TabShots also places an icon onto the browser's toolbar that changes color in response the amount of changes on the currently focused tab. The icon serves as an unobtrusive security layer that will remain in view should the overlay be removed.

Finally, there is also an intended component for blacklisting sites that have confirmation of tabnabbing attempts. The developers have created an optaionl server-side component which can pull reported URLs from individual users, and add the confirmed URLs to a database. If the user decides that the current page is attempting tabnabbing and provides their explicit approval, the application can be signaled to send the accused page's URL, the image before the user switched tabs, and the image of the page after the user switched tabs. The server-side application uses its own automated comparison process and the further verification of a human analyst to decide if the page is indeed a phishing page, and whether the URL should be added to a subscription model blacklist.

2.2 TabSol

2.3 NoTabNab

3 Proposal and Methodology

4 Results

5 Conclusion

References

[1] Lieven Desmet Wouter Joosen Philippe De Ryck, Nick Nikiforakis. Tabshots: Client-side detection of tabnabbing attacks, 2013. 978-1-4503-1767-2.