Web Security

Prof. Yinzhi Cao

Fall 2019

Homework 3: Chrome extension for detecting Tabnabbing Attacks

Project due on November 21st in Blackboard

Tabnabbing is a form of phishing where malicious websites exploit the lack of focus of their website to completely change their look and feel to imitate the log-in page of a benign site. The key idea of tabnabbing is that users are trained to carefully inspect a website when they first open it, but once it's open, it is lost in a "sea of tabs". When users come back to the open site (that now may look like the login page of Paypal or GMail) they do not inspect the browser URL but assume that Gmail/Paypal has logged them out and proceed to log-in again, thereby leaking their credentials to the attacker.

You can see a video of the attack here and read more about it here (at the time of this writing, this link tries to demonstrate the attack when you lose focus but fails because of a non-existing image. You can read the entire thing as long as you don't switch tabs/windows while you are doing it.)

In this project, you are called to create a Chrome extension which will alert users when a tabnabbing attack has taken place by detecting that the tab that the user is looking at is different than it was before it "lost" focus.

High-level Description of steps:

1. While a user is browsing a webpage, take screenshots on regular intervals, always keeping the last one.
2. Detect the change to a new tab (loss of focus).
3. When a user returns to the tab, take a fresh screenshot and compare the two.
4. Highlight the changes on the page and provide a color coding in the task bar to alert the user of potential changes

Notes

- There is a 2013 paper proposing the exact same system which is available here. Even though I encourage you to read through it to understand the system, I am encouraging you NOT to look at the source code linked from that paper. If you do, you may end up developing

something very similar which will be flagged by our plagiarism-detecting systems.

- The comparison algorithm that you use should do the comparison locally via JavaScript. You are allowed to use existing libraries, such as, Resemble.js
- You should be able to identify the parts of the page that have changed versus the ones have remained the same and highlight the relevant changes on the page. (simple approach: Split the page in squares, compare each square with its previous version, color the modified squares.)
- You should only give minimum permissions to the Chrome extension.